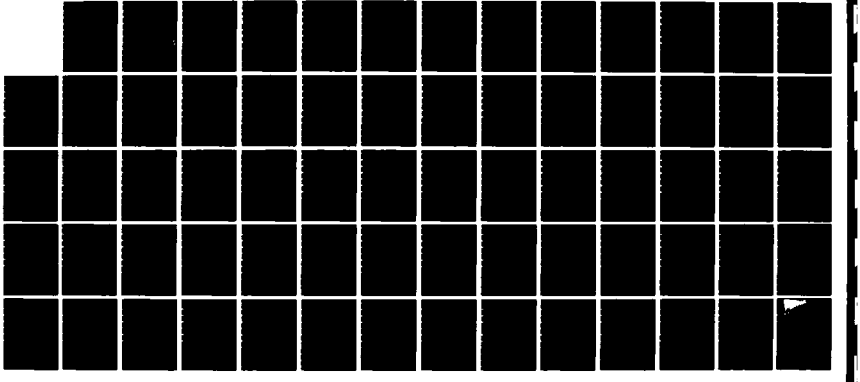
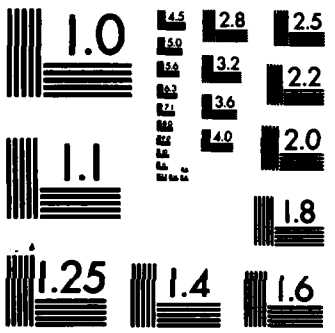
AD-A140 171 HEURISTIC ALGORITHMS FOR SOLVING TWO DIMENSIONAL LOADING PROBLEMS(U) FLORIDA UNIV GAINESVILLE DEPT OF INDUSTRIAL AND SYSTEMS ENGINEERING M A WHITE MAR 81 RR-81-3 F01600-79-D-0146 F/G 12/1 NL

1/1

UNCLASSIFIED





HEURISTIC ALGORITHMS FOR SOLVING TWO DIMENSIONAL LOADING PROBLEMS

RESEARCH REPORT NO. 81-3

BY

MICHAEL A. WHITE, CAPTAIN, USAF

MARCH, 1981

DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING
UNIVERSITY OF FLORIDA
GAINESVILLE, FLORIDA 32611

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

THIS RESEARCH WAS SUPPORTED IN PART BY THE U.S. AIR FORCE, UNDER CONTRACT NUMBER SCEEE/AFLMC/80-6/F01600-79-D0146 AND BY THE OFFICE OF NAVAL RESEARCH, UNDER CONTRACT NUMBER N00014-76-C-0096.

THE FINDINGS OF THIS REPORT ARE NOT TO BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE AIR FORCE OR DEPARTMENT OF THE NAVY POSITION, UNLESS SO DESIGNATED BY OTHER AUTHORIZED DOCUMENTS.

TABLE OF CONTENTS

A-1

HEURISTIC ALGORITHMS FOR SOLVING TWO DIMENSIONAL LOADING PROBLEMS

ABSTRACT

THE LOADING PROBLEM INVOLVES THE ALLOCATION OF 'N' BOXES, EACH HAVING A SPECIFIC LENGTH, WIDTH AND HEIGHT, TO A PALLET OF SPECIFIC DIMENSIONS. MUCH WORK HAS BEEN DONE ON SOLVING THE MANY SPECIAL CASES OF THIS PROBLEM WHERE ALL BOXES ARE OF THE SAME RECTANGULAR SIZE AND/OR WITH THE RESTRICTION THAT THE EDGE OF THE RECTANGLES FOLLOW A "GUILLOTINE CUT" FROM ONE EDGE OF THE PALLET TO THE OTHER; THE TWO DIMENSIONAL CUTTING STOCK PROBLEM. HOWEVER, THE GENERALIZED PROBLEM OF DIFFERENT SIZED BOXES BECOMES VERY DIFFICULT TO SOLVE. A COMMON PROBLEM FOR THE U.S. AIR FORCE IS THE TRANSPORTATION OF EQUIPMENT IN A LARGE NUMBER OF BOXES, EACH OF DIFFERENT SIZES. THESE BOXES WOULD BE LOADED ONTO PALLETS FOR SUBSEQUENT PLACEMENT ON TRANSPORT AIRCRAFT. GENERATING THE METHODS AND INSTRUCTIONS FOR LOADING THE PALLETS IS ROUTINELY ACCOMPLISHED MANUALLY AND RELYS HEAVILY ON THE EXPERIENCE OF THE TRANSPORTATION PERSONNEL TO DETERMINE LOADING PATTERNS THAT WILL PRODUCE GOOD PALLET USAGE. THEREFORE, UTILIZING A COMPUTER TO GENERATE THE LOADING PROCEDURE WOULD BE OF CONSIDERABLE PRACTICAL BENIFIT IN REDUCING LOADING TIME. THIS REPORT PRESENTS SEVERAL HEURISTRIC ALGORITHMS FOR SOLVING LARGE TWO DIMENSIONAL LOADING PROBLEMS. THE OBJECTIVE OF THE ALGORITHMS IS TO MAXIMIZE THE RATIO OF AREA USED TO THE TOTAL PALLET AREA. THE PROCEDURES EMPLOY DYNAMIC PROGRAMMING AND A "STACKING PROCEDURE" FOR POSITIONING BOXES ON THE PALLET. A TOTAL OF FIVE ALGORITHMS ARE USED, EACH HAVING A DIFFERENT PALLET TO BOX ORIENTATION AND USING VARIOUS COMBINATIONS OF THE "STACKING PROCEDURE". SOLUTIONS TO ALL FIVE ALGORITHMS ARE COMPARED TO FIND THE BEST SOLUTION BASED ON TOTAL LOADED AREA. THE ALGORITHMS ALSO PROVIDE THE COORDINATES OF THE LOADED BOXES ON THE PALLET TO ASSIST IN THE ACTUAL POSITIONING OF THE ITEMS.

PROBLEM

1. CONSIDER THE FOLLOWING PROBLEM; ALLOCATE A SET OF "N" BOXES, EACH HAVING A SPECIFIED LENGTH, WIDTH AND HEIGHT, TO A PALLET OF LENGTH "L" AND WIDTH "W".

FACTORS BEARING ON THE PROBLEM.

2. THE STATED PROBLEM CONSISTS OF FINDING THE BEST LOADING PATTERN OF "N" BOXES WHICH YIELDS AN EFFICIENT UTILIZATION OF THE PALLET. THE APPROACH TAKEN TO SOLVE THIS PROBLEM WAS TO DEVELOP A SERIES OF HEURISTIC ALGORITHMS, EACH OF WHICH VARY THE LOADING PATTERN OF THE BOXES AND THEN SELECT THE BEST SOLUTION.

SINCE THESE HEURISTICS ARE ESSENTIALLY A TRIAL AND ERROR PROCEDURE THEIR FORMULAS BECOME VERY LABORIOUS AND TIME CONSUMING TO IMPLEMENT. THUS, DEVELOPMENT OF COMPUTER CODES TO SIMULATE THE LOADING ALGORITHMS BECOMES A NECESSITY. FOR THIS PROBLEM FORTRAN LANGUAGE WAS USED TO DEVELOP THE CODES, ON A PDP-11/34.

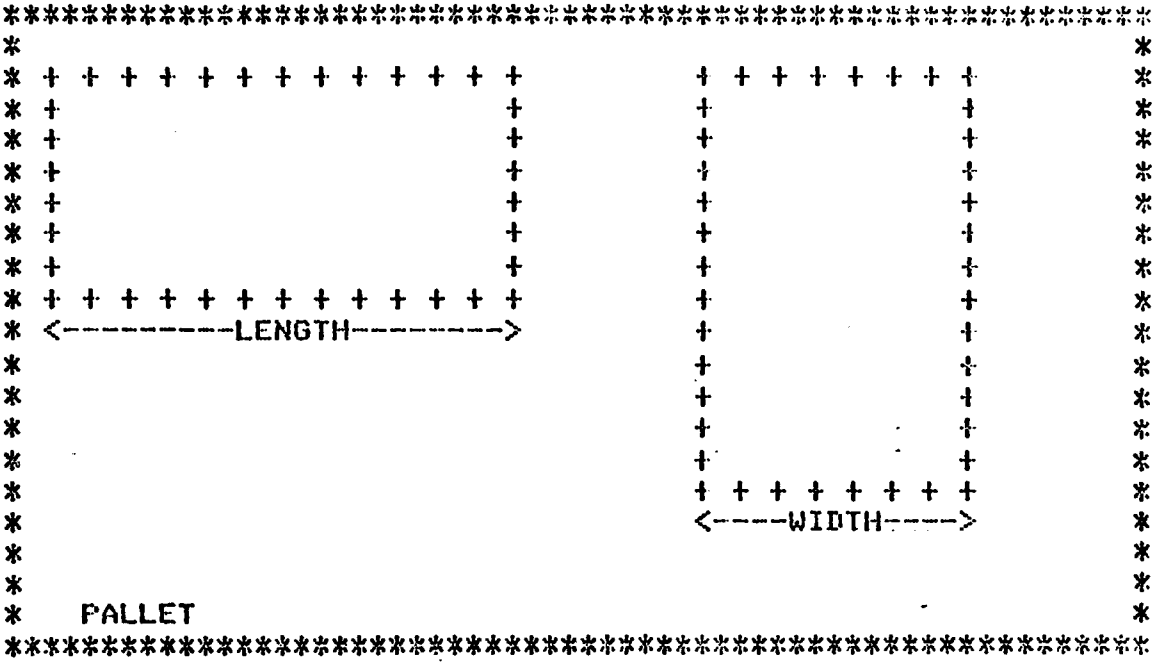
ASSUMPTIONS

3. TO OBTAIN INITIAL SOLUTIONS, SEVERAL IMPORTANT ASSUMPTIONS WERE MADE WHICH CONSIDERABLY SIMPLIFIED THE OVERALL PROBLEM. MANY OF THESE ASSUMPTIONS HAVE BEEN, OR WILL BE ADDRESSED IN OTHER ALGORITHMS WHICH CAN BE USED IN CONJUNCTION WITH THE SOLUTIONS PRESENTED HERE.

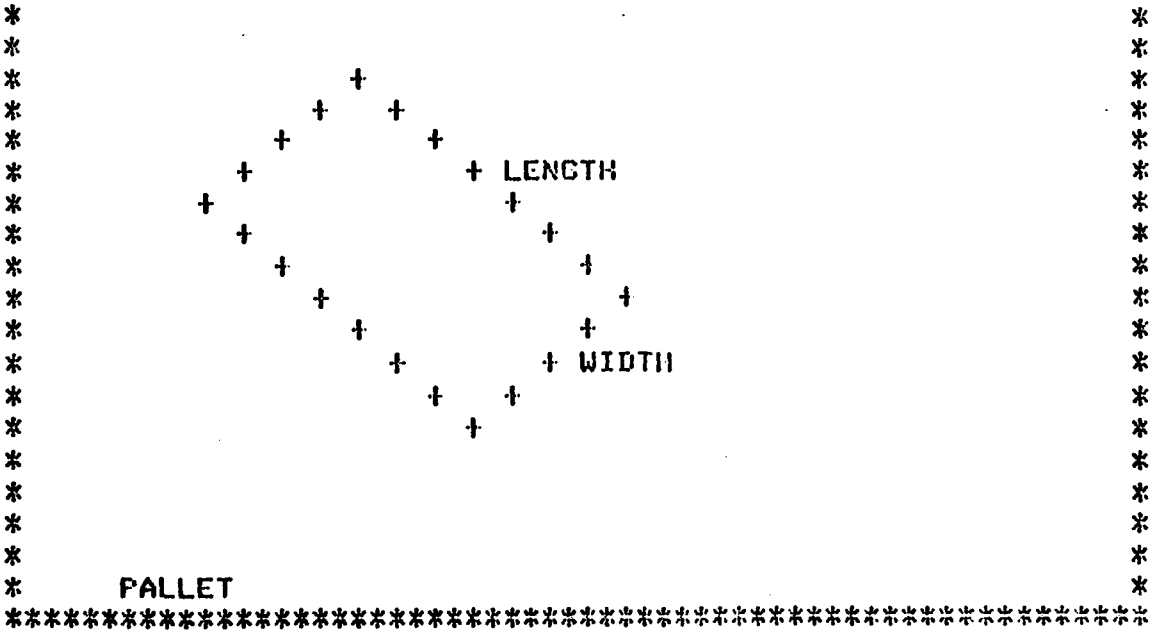
(A) THE ORIENTATION OF EACH BOX, IN TERMS OF ITS LENGTH, WIDTH AND HEIGHT HAS BEEN FIXED PRIOR TO ATTEMPTING TO LOAD THE BOX ONTO THE PALLET.
(B) ALL BOXES WILL BE CONSIDERED IN A "THIS-END-UP" ORIENTATION WITH THE HEIGHT DIMENSION CONSIDERED TO BE 'UP'.
(C) THE WEIGHT OF THE BOXES WILL NOT BE CONSIDERED WHEN POSITIONING THEM ON THE PALLET.
(D) A BOX MUST ONLY BE POSITIONED WITH ITS LENGTH OR WIDTH PARALLEL TO THE LENGTH OF THE PALLET(SEE FIG. 1).
(E) THE SET OF BOXES CONSIDERED FOR LOADING AT ANY ONE TIME WILL BE LIMITED TO NO MORE THAN 30.

THE FIRST TWO ASSUMPTIONS RESTRICTS THE SOLUTION TO USING ONLY THE LENGTH AND WIDTH OF THE BOXES WHEN ASSIGNING THEM TO POSITIONS ON THE PALLET (SEE FIG. 2).

THE THIRD ASSUMPTION ELIMINATES THE REQUIREMENT FOR CONSIDER-

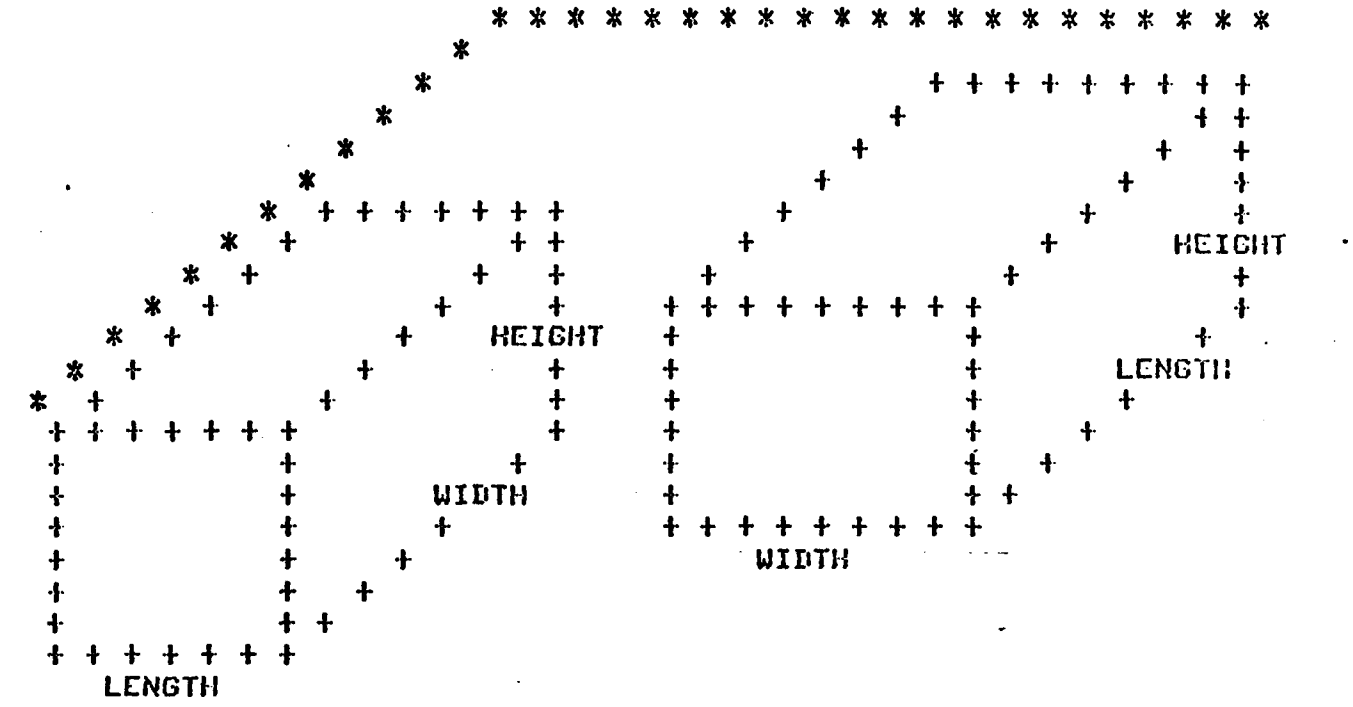


ACCEPTABLE
PALLET-BOX ORIENTATION

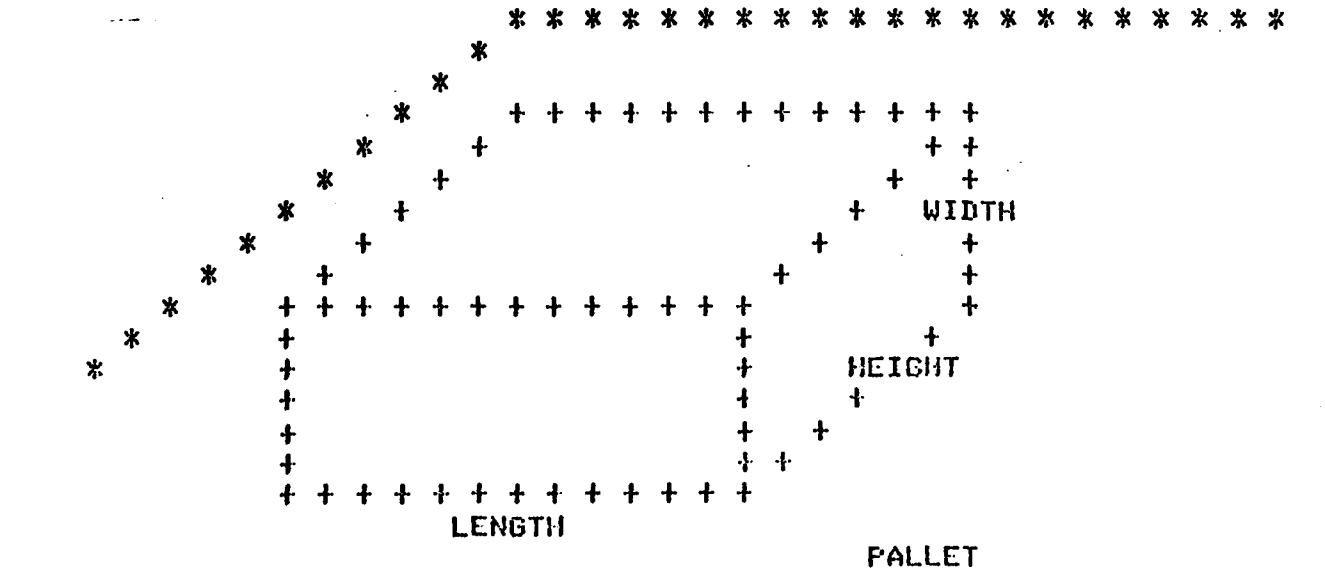


UNACCEPTABLE
PALLET-BOX ORIENTATION

FIGURE 1



ACCEPTABLE BOX ORIENTATION



UNACCEPTABLE BOX ORIENTATION

FIGURE 2

ING WEIGHT STABILITY IN ANY SOLUTION.

FINALLY, THE LAST ASSUMPTION IS BASED ON AN EXAMINATION OF THE COMPLETE DATA SET CONTAINING THE DIMENSIONS OF ALL POSSIBLE BOXES THAT COULD BE LOADED ON THE PALLET AND ON THE PALLET SIZE OF 104 INCHES IN LENGTH AND 84 INCHES IN WIDTH.

CRITERIA

4. THERE ARE OF COURSE SEVERAL STANDARDS BY WHICH THE ALGORITHMS CAN BE EVALUATED TO DETERMINE THE OPTIMAL LOADING ROUTINE. HOWEVER, IN LIGHT OF THE ASSUMPTIONS THAT WERE MADE, THE OBVIOUS CRITERIA WOULD BE FIRST, THE TOTAL PALLET AREA USED, AND SECONDLY, THE NUMBER OF BOXES LOADED.

DISCUSSION

5. IN DEVELOPING THESE SOLUTION ALGORITHMS THE UNDERLYING GOAL WAS TO SIMULATE THE ACTUAL STEPS THAT WOULD BE TAKEN BY AN INDIVIDUAL WHO WAS FACED WITH OF LOADING A PALLET WITH 'N' BOXES SO AS TO MAXIMIZE THE TOTAL AREA USED.

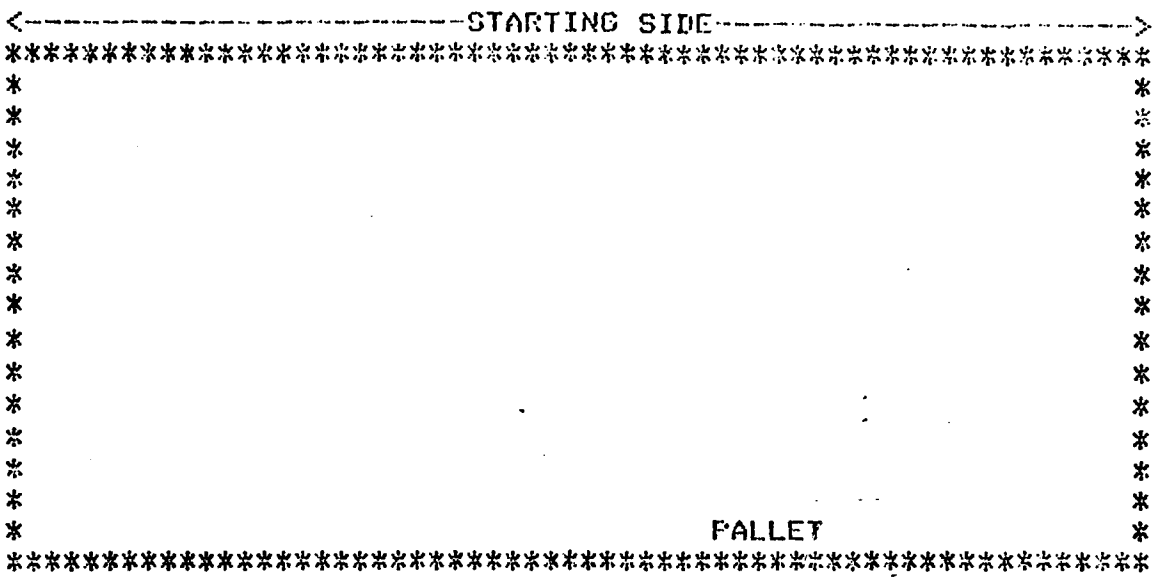
FACED WITH THIS PROBLEM AND GIVEN THE ASSUMPTIONS PRESENTED IN PARA. 3, AN INDIVIDUAL COULD CHOOSE FROM SEVERAL DIFFERENT PROCEDURES IN ATTEMPTING TO ACHIEVE AN ACCEPTABLE LOAD. LOADING FROM THE PERIMETER INWARD IS ONE POSSIBILITY. ALSO, DIVIDING THE PALLET AREA INTO SMALLER SECTION AND LOADING EACH OF THE SECTIONS COULD BE TRIED. HOWEVER, THE PROCEDURE FOLLOWED IN THE PROPOSED SOLUTIONS IS WHAT CAN BE TERMED THE "STACKING ALGORITHM". THIS "STACKING ALGORITHM" WOULD TYPICALLY PROCEED IN THE FOLLOWING MANNER.

INITIAL LOAD

STEP(1): SELECT EITHER THE LENGTH OR WIDTH OF THE PALLET AS THE 'STARTING SIDE' TO BEGIN LOADING THE BOXES. THAT IS, THE PALLET WOULD BE ORIENTATED EITHER LENGTH-WISE OR WIDTH-WISE (SEE FIG. 3).

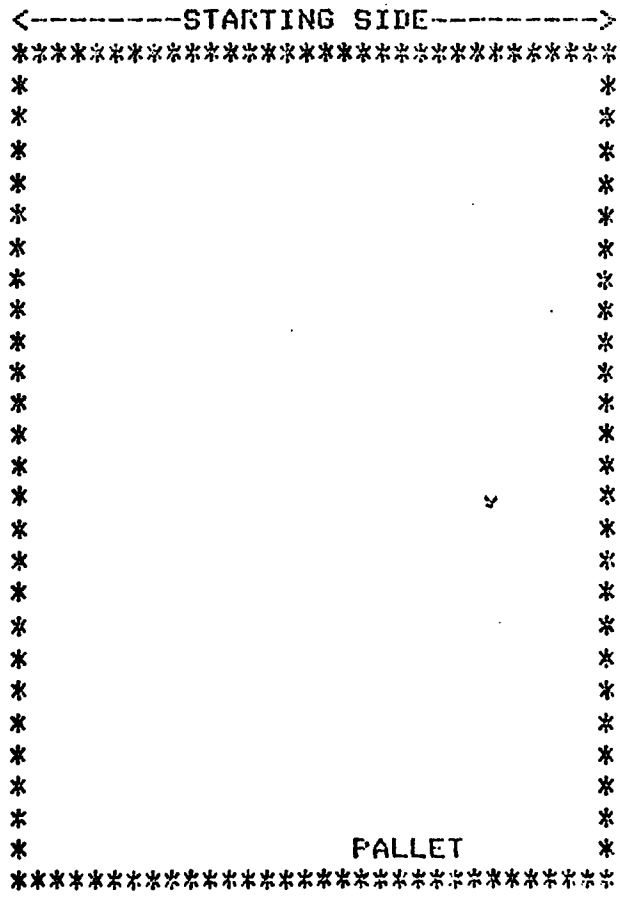
STEP(2): SELECT A REFERENCE OR STARTING POINT, SAY THE UPPER LEFT HAND CORNER. THIS IS WHERE THE FIRST BOX WILL BE POSITIONED (SEE FIG. 3).

STEP(3): THEN FROM A SET OF NO MORE THAN 30 BOXES, ATTEMPT TO LOAD AS MANY BOXES ALONG THE "STARTING SIDE", BEGINNING WITH THE UPPER LEFT CORNER OF THE FIRST BOX COINCIDING WITH THE PALLET REFERENCE POINT. ALL BOXES WOULD BE ORIENTATED EITHER WITH ITS LENGTH PARALLEL OR PERPENDICULAR TO THE STARTING SIDE OF THE PALLET (SEE FIG. 4). THIS BOX ORIENTATION WILL BE MAINTAINED FOR ALL REMAINING LOADS.



PALLET

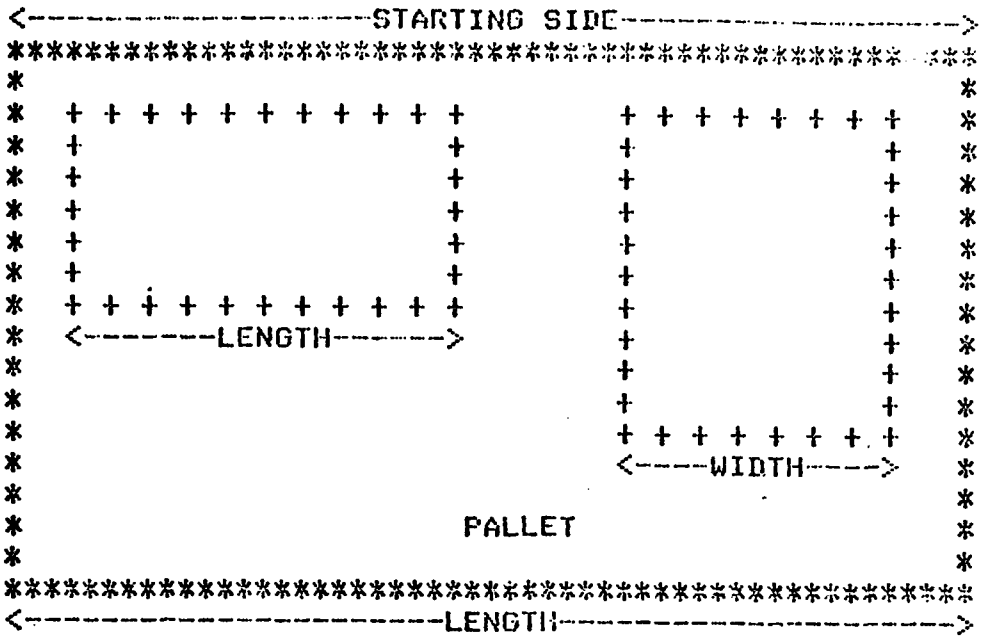
PALLET LENGTH ORIENTATION

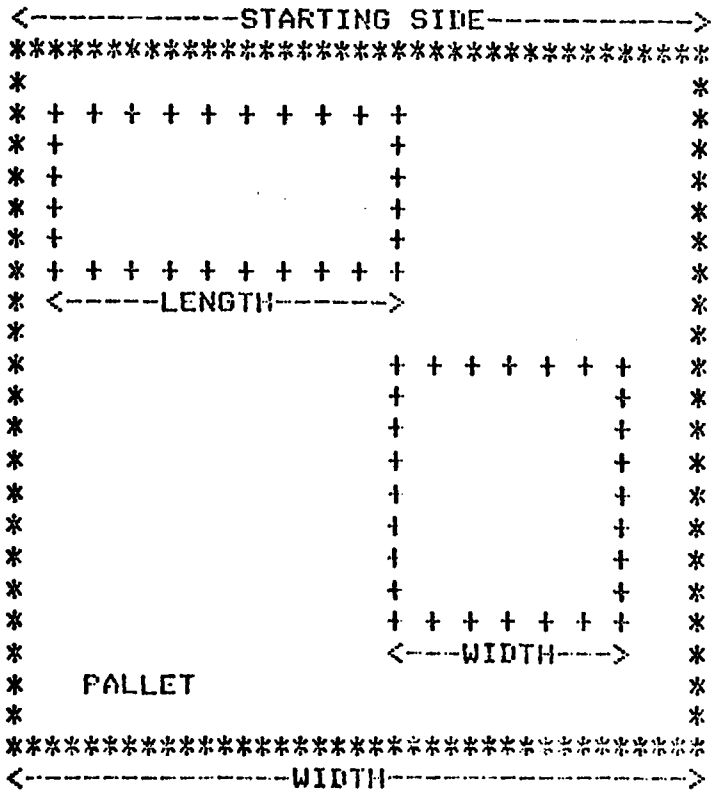


PALLET

PALLET WIDTH ORIENTATION

FIGURE 3





ACCEPTABLE
BOX ORIENTATION

FIGURE 4

STEP(4): THE LOADING OF BOXES WOULD CONTINUE UNTIL NO MORE BOXES WILL FIT IN THE REMAINING SPACE, 'S', ALONG THE "STARTING SIDE".

STEP(5): EVERY COMBINATION OF BOXES WOULD BE TRIED SO AS TO MINIMIZE THE REMAINING SPACE, 'S'. THE OPTIMAL COMBINATION OF BOXES WOULD THEN BE CHOOSEN AS THE INITIAL LOAD.

AT THIS POINT ONE MUST REALIZE THAT THE INITIAL LOADING PATTERN JUST FOUND DOES NOT CONSIDER THE SECOND DIMENSION OF THE BOXES BEING LOADED. THEREFORE IT, WILL DEPEND GREATLY ON THE DISTRIBUTION OF THE LENGTHS AND WIDTHS OF THE SET OF BOXES TO BE LOADED, AS TO WHETHER THE INITIAL LOAD IS "UNIFORM" OR "UNEVEN".

LET US EXAMINE THE MEANING OF AN "UNIFORM" OR "UNEVEN" LOAD, AND THE EFFECT EACH HAS ON THE REMAINING BOXES LOADED.

AS AN EXAMPLE, CONSIDER THE SET OF 30 BOXES THAT HAVE ONLY TWO DIFFERENT BOX LENGTHS AND WIDTHS (SEE COL. A, FIG. 5). AN INITIAL LOADING PATTERN, WITH THE BOX LENGTH PARALLEL TO THE "STARTING SIDE", WILL LOOK LIKE FIGURE 6. THIS LOAD WILL HAVE A VERY "UNIFORM" APPEARENCE BECAUSE OF THE SIMILAR BOX WIDTHS. THAT IS, THE BOXES WOULD NOT PRODUCE A "STEP" PATTERN.

NOW COMPARE THE LOAD IN FIGURE 6 TO THE INITIAL LOAD CREATED BY THE SET OF 30 BOXES SHOWN IN COL. B, FIGURE 5, WERE THE RANGE OF LENGTHS AND WIDTHS IS VERY LARGE. THIS LOAD PRESENTS A VERY "UNEVEN" PATTERN BECAUSE OF THE DIFFERENT BOX WIDTHS (SEE FIG. 7). THIS DIFFERENCE IN INITIAL LOAD PATTERNS WILL DICTATE THE REMAINING LOADING PROCEDURE.

LET'S FIRST TAKE A LOOK AT THE CASE WHERE THERE WAS AN "UNEVEN" PATTERN TO THE INITIAL LOAD. THE LOADING OF THE REMAINING BOXES WOULD CONTINUE AS FOLLOWS.

UNEVEN INITIAL LOAD

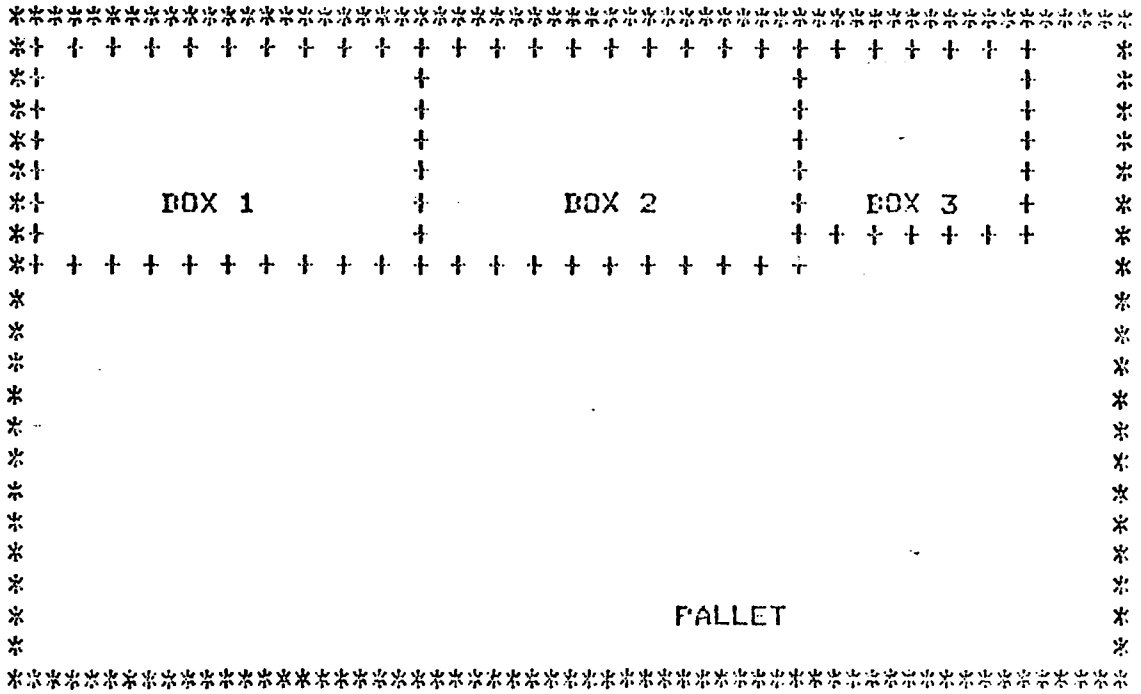
STEP(6): BEGINNING WITH THE FIRST BOX IN THE INITIAL LOAD, BOXES ARE POSITIONED ALONG THE UNDERSIDE OF THIS BOX IN ONE OF TWO METHODS;

(A) BY THE PROCEDURE DESCRIBED IN THE INITIAL LOADING SECTION, STEPS (1) THRU (5), WITH THE LENGTH OF THE UNDERSIDE OF THIS FIRST BOX TAKING THE PLACE OF THE PALLETS STARTING SIDE (SEE FIG. 8)

(B) FROM THE REMAINING BOXES CHOOSE THE LONGEST(OR IF THE BOX WIDTH ORIENTATION IS USED, THE WIDEST) BOX THAT WILL FIT UNDER THE FIRST BOX IN THE INITIAL LOAD. IF THERE IS STILL SPACE REMAINING AFTER THIS BOX IS POSITIONED THEN SELECT THE NEXT LONGEST(OR WIDEST) BOX THAT WILL FIT IN

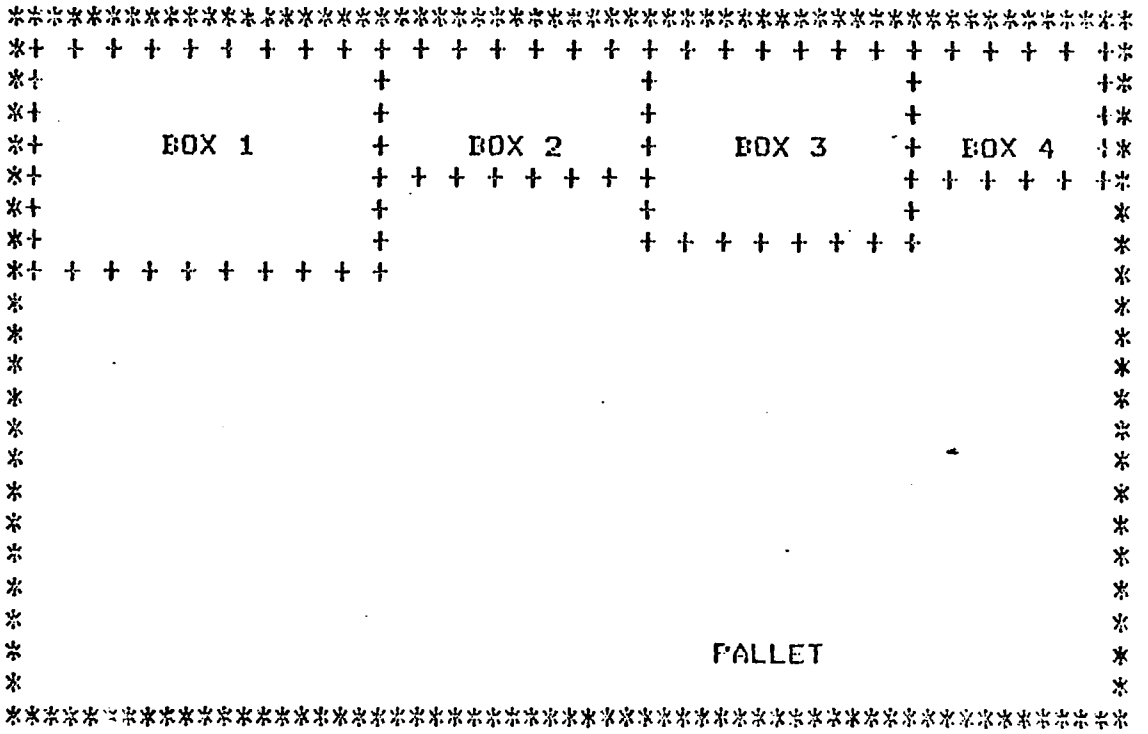
LENGTH	WIDTH	LENGTH	WIDTH
99	52	34	20
97	25	34	20
78	5	34	20
68	33	34	20
68	33	34	20
67	33	34	20
67	34	34	20
67	34	34	20
62	6	34	20
60	30	34	20
57	33	34	20
43	19	34	20
43	19	34	20
43	19	34	20
40	4	34	20
38	38	34	20
37	21	34	20
37	21	34	20
36	36	30	18
35	35	30	18
33	27	30	18
31	20	30	18
31	16	30	18
31	17	30	18
30	24	30	18
27	24	30	18
21	21	30	18
20	18	30	18
20	19	30	18
8	4	30	18
COL. A		COL. B	

FIGURE 5



UNIFORM INITIAL LOAD

FIGURE 6



UNEVEN INITIAL LOAD

FIGURE 7

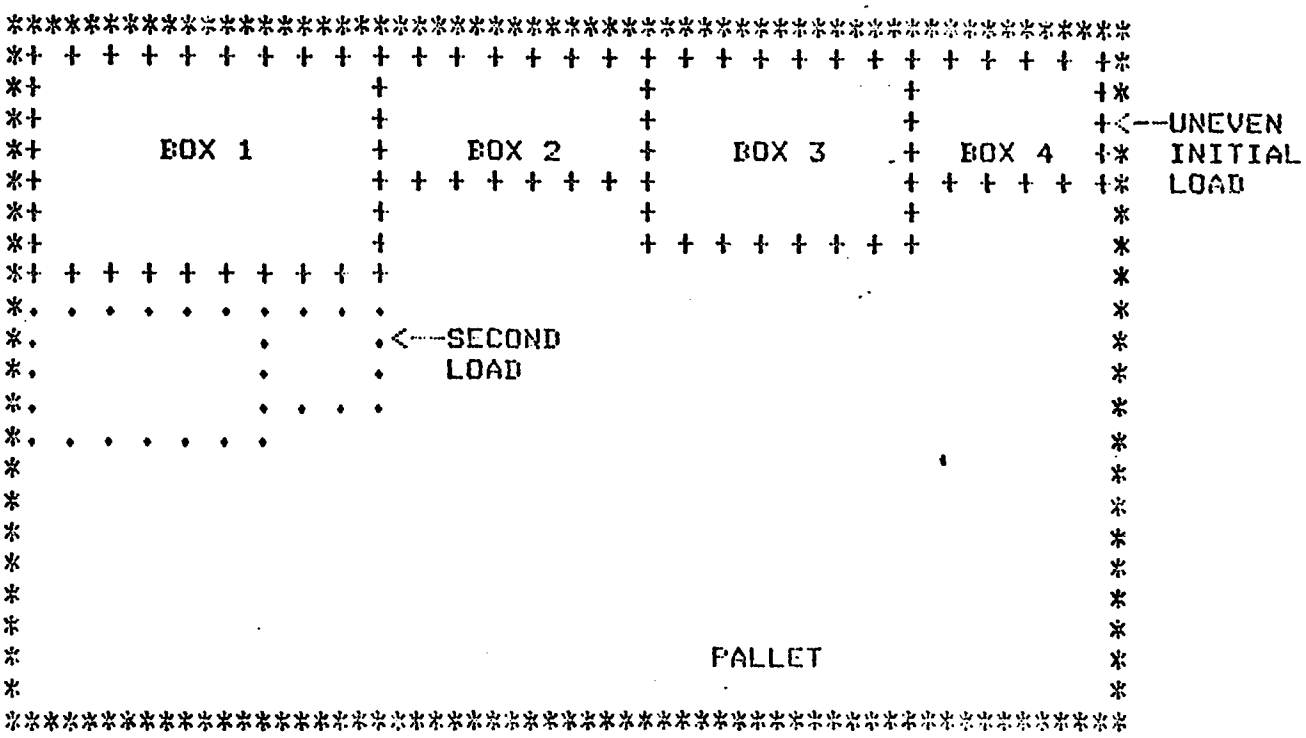


FIGURE 8

THIS SPACE (SEE FIG. 9).

STEP(7): THE PROCEDURE IN STEP(6), EITHER (A) OR (B) WOULD THEN BE REPEATED FOR EACH BOX OF THE SECOND LOAD AND FOR EACH SUBSEQUENT LOAD PROCEDING DOWN THE SIDE OF THE PALLET UNTIL NO FURTHER BOXES CAN BE LOADED.

STEP(8): ONCE THE COMPLETED SECOND LOAD IS IN POSITION, THE THIRD LOAD WILL BEGIN UNDER THE SECOND BOX OF THE INITIAL LOAD. THE SAME PROCEDURE CHOOSEN IN STEP(6) WILL BE USED.

WHAT STEPS (6) THRU (8) ARE DOING IS "STACKING" SUCESSIVE BOXES UNDER EACH OTHER, HENCE THE NAME "STACKING ALGORITHM". THIS STACKING PROCEDURE WILL CONTINUE UNTIL ALL BOXES ARE USED OR NO MORE BOXES WILL FIT IN THE REMAINING SPACE OF THE PALLET.

NOW CONSIDER THE "UNIFORM" INITIAL LOAD. HERE AN INTERMEDIATE STEP CAN BE PERFORMED, BEFORE THE "STACKING" BEGINS.

UNIFORM INITIAL LOAD.

STEP(6'): THIS INTERMEDATE STEP IS TO REPEAT STEPS (1) THRU (5) USING THE REMAINING UNLOADED BOXES WITH THE PALLET "STARTING SIDE" LENGTH REPLACED BY THE SUM OF THE LENGTHS(OR WIDTHS) OF THE BOXES IN THE INITIAL LOAD (SEE FIG.10).

IF THIS SECOND LOAD ALSO HAS A "UNIFORM" PATTERN THE PROCEDURE IS REPEATED. HOWEVER, IF THE SECOND LOAD IS "UNEVEN" THEN STEPS(6) THRU (8) ARE IMMEDIATELY FOLLOWED.

UNIFORM LOAD ASSUMPTION

AN IMPORTANT MODIFICATION OF THE ABOVE PROCEDURE MUST NOW BE INVESTIGATED.

WHEN CONSIDERING THE STEPS TO FOLLOW WHEN CONSTRUCTING THE SECOND LOAD, OR ANY SUBSEQUENT LOAD, EXAMINATION OF A RESULTING "UNEVEN" LOAD MUST BE CONDUCTED. CAN THIS "UNEVEN" PATTERN BE ASSUMED TO BE A "UNIFORM" LOAD? THIS DETERMINATION WILL BE HIGHLY SUBJECTIVE. FOR EXAMPLE, IF AN INITIAL LOAD PATTERN HAS BOX WIDTH DIFFERENCES OF NO MORE THAN SAY 4 INCHES BETWEEN ANY TWO BOXES THE LOAD COULD BE CONSIDERED "UNIFORM". ON THE OTHER HAND, THE REQUIREMENT MAY BE SET AT NO MORE THAN 2 INCHES. FOR THIS PROBLEM ASSUME A 4 INCH DIFFERENTIAL IS REQUIRED.

AS AN EXAMPLE, CONSIDER THE INITIAL LOAD SHOWN IN FIGURE 11. THERE ARE 3 BOXES WITH VARIOUS WIDTHS, AN UNEVEN LOAD. EXAMINATION OF THESE WIDTHS INDICATES THAT THE DIFFERENCE BETWEEN ANY TWO BOX WIDTHS IS LESS THAN 4 INCHES. IN THIS CASE AN IMAGINARY LINE CAN BE DRAWN CREATING AN "UNIFORM" LOADING PATTERN (DASHED LINE). NOW INSTEAD OF IMMEDIATELY PROCEDING TO STEP(6) WHERE "STACKING" BEGINS, STEPS(1) THRU (5) CAN BE REPEATED, WITH

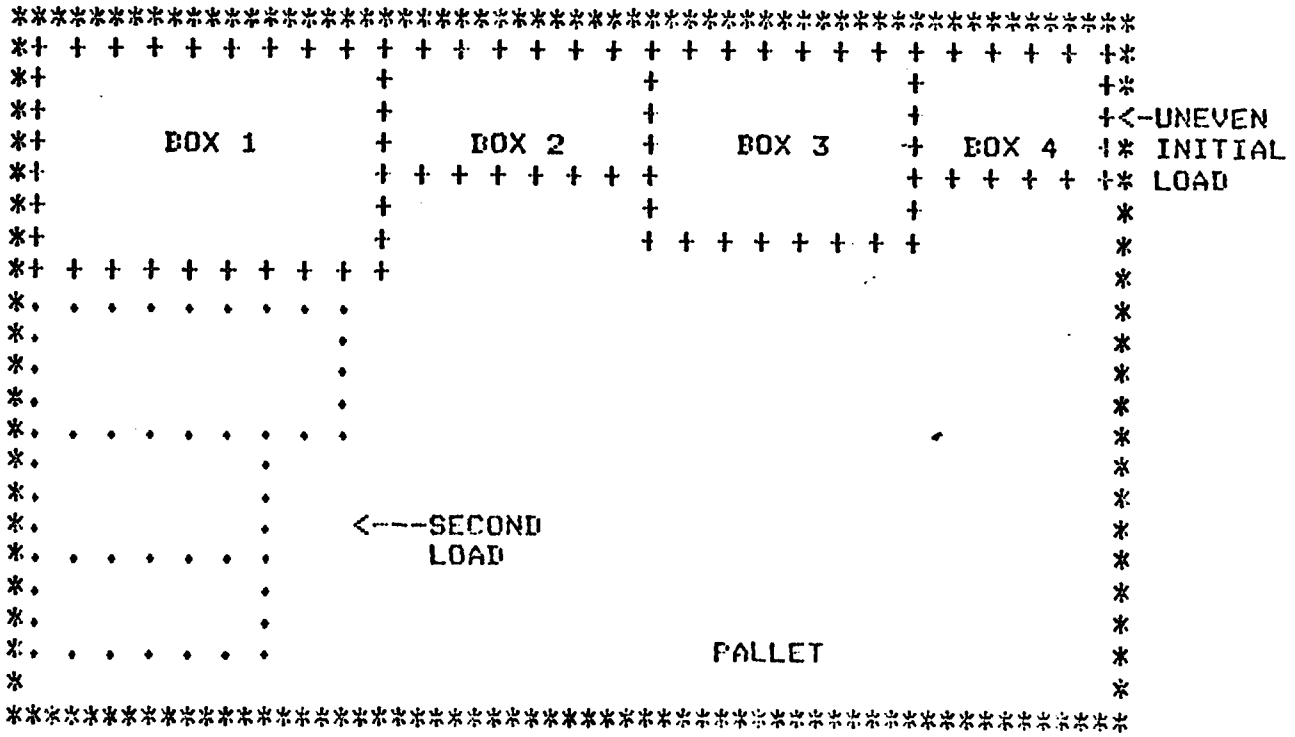


FIGURE 9

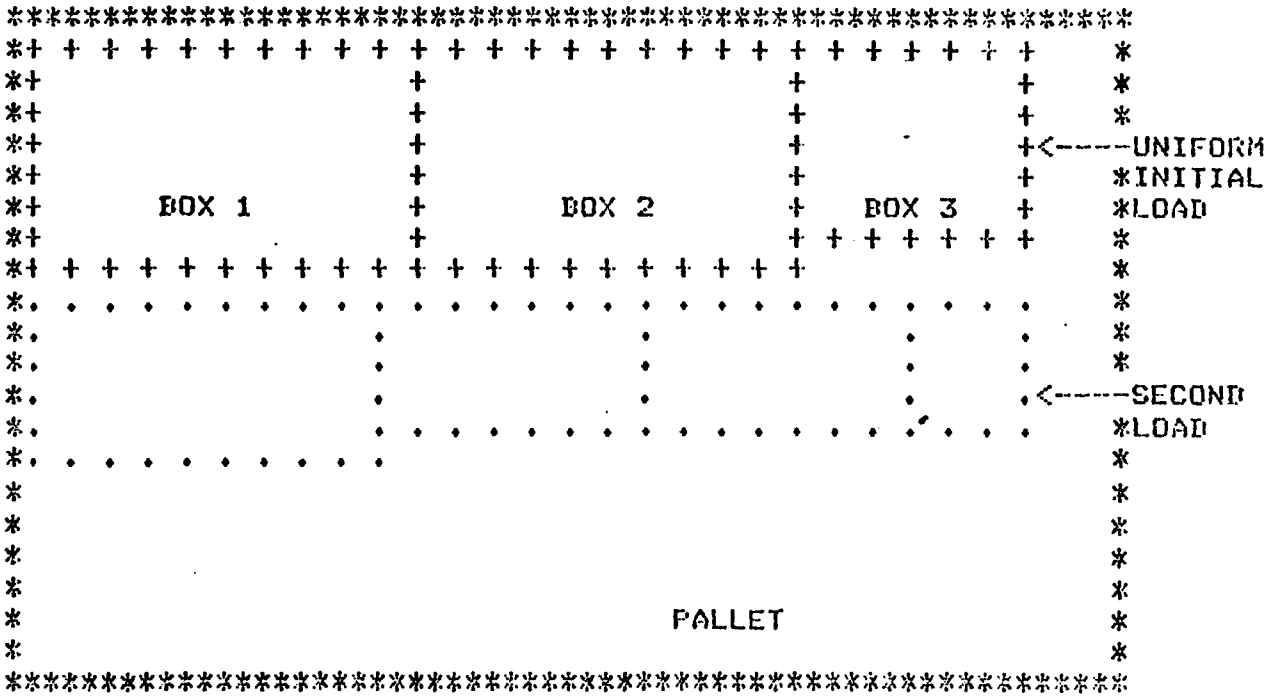
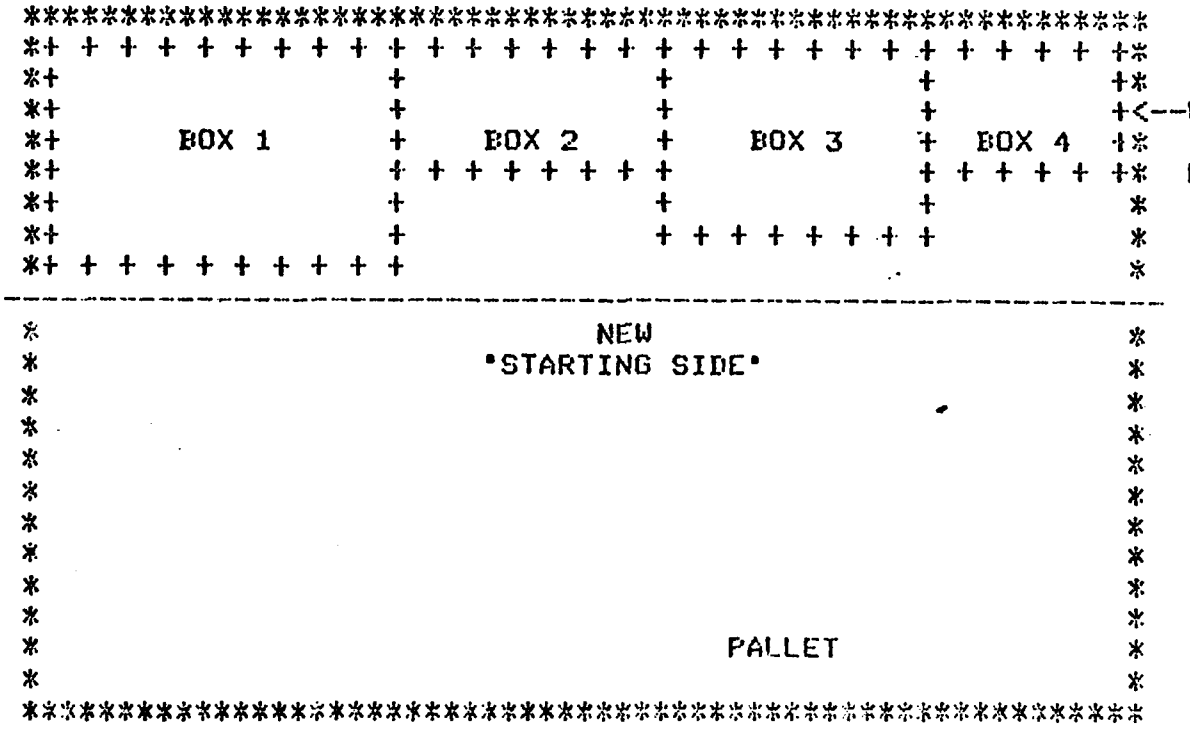


FIGURE 10



UNIFORM LOAD ASSUMPTION
FOR AN ENTIRE INITIAL LOAD

FIGURE 11

THE DASHED LINE TAKING THE PLACE OF THE PALLET "STARTING SIDE". THAT IS, STEP(6') CAN BE USED BEFORE THE "STACKING" STEPS. THIS PROCEDURE CAN ALSO BE APPLIED TO ANY SUBSEQUENT LOAD.

ALSO, THIS PROCEDURE CAN BE EXTENDED TO PORTIONS OF A PARTICULAR LOAD. IF AN INTERMIDIATE LOAD APPEARS AS IN FIGURE 12 'BOX A' AND 'BOX B' COULD BE CONSIDERED AS ONE BOX WITH A COMBINED LENGTH OF X+Y. THIS IS BECAUSE THE DIFFERENCE IN BOX WIDTHS IS LESS THAN 4 INCHES. HOWEVER, 'BOX C' CAN NOT BE INCLUDED IN THIS COMBINATION BECAUSE ITS WIDTH IS MUCH SMALLER THAN 'BOX B'.

THE IMPORTANT POINT TO REMEMBER IS THAT BECAUSE THE UNUSED SPACE ABOVE THIS IMAGINARY LINE WILL BE WASTED PALLET SPACE, IT IS CRITICAL TO DETERMINE THE "BEST" BOX DIFFERENCE TO USE IN DETERMINING IF A "UNIFORM" PATTERN CAN BE ASSUMED. AGAIN THIS SELECTION IS "VERY" SUBJECTIVE. ANALYSIS OF THE DIMENSIONS OF THE AVAILABLE BOXES MAY GIVE SOME INSIGHT INTO THE PROPER DIFFERENTIAL REQUIRED TO ENSURE A MINIMUM OF UNUSED PALLET AREA.

COMPUTER MODEL

6. ALTHOUGH THE GENERAL PROCEDURE JUST DESCRIBED IS FAIRLY STRAIGHT FORWARD, WHEN ATTEMPTED MANUALLY, THERE CAN BE A CONSIDERABLE AMOUNT OF WORK INVOLVED IN FINDING THE BEST "FINAL" LOADING PATTERN.

THE COMPUTER CODES FOR THE HEURISTIC ALGORITHMS ARE DIVIDED INTO TWO CLASSES. BOTH CLASSES BASICALLY CORRESPOND TO THE LOADING PROCEDURE OUTLINED IN PARA. 5. THE ONLY DIFFERENCE BETWEEN THE TWO IS THAT THE FIRST CLASS OF CODES USES THE STACKING PROCEDURE DESCRIBED IN STEP (6)(A), WHERE AS THE SECOND USES THE METHOD OF STEP(6)(B). BOTH USE THE "UNIFORM LOAD" ASSUMPTION PROCEDURE.

ALSO, WITH THE FIRST TYPE OF PROCEDURE THERE ARE FOUR DIFFERENT CODES, CALLED "MODULES". EACH MODULE USES A DIFFERENT PALLET-BOX ORIENTATION. THESE FOUR MODULE ORIENTATIONS ARE;

(A) PALLET LENGTH PARALLEL TO BOX WIDTH,
(B) PALLET WIDTH PARALLEL TO BOX WIDTH,
(C) PALLET WIDTH PARALLEL TO BOX LENGTH, AND
(D) PALLET LENGTH PARALLEL TO BOX LENGTH.

SEE FIGURE 13 FOR A GRAPHICAL REPRESENTATION OF THESE FOUR ORIENTATIONS.

THE SECOND CLASS OF PROCEDURES CONTAINS ONLY ONE BASIC CODE, WERE THE PALLET LENGTH IS PARALLEL TO THE BOX LENGTH. HOWEVER, THIS SECOND CLASS OF PROCEDURES COULD BE EXTENDED TO INCLUDE ALL FOUR ORIENTATIONS, BUT WAS NOT ATTEMPTED

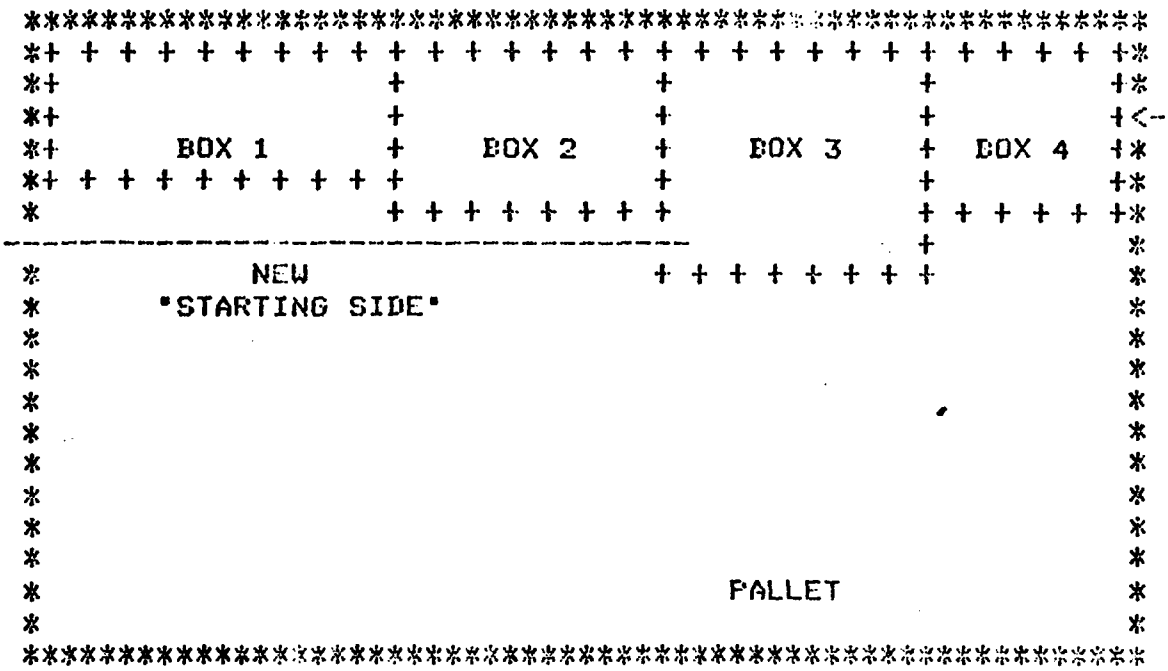
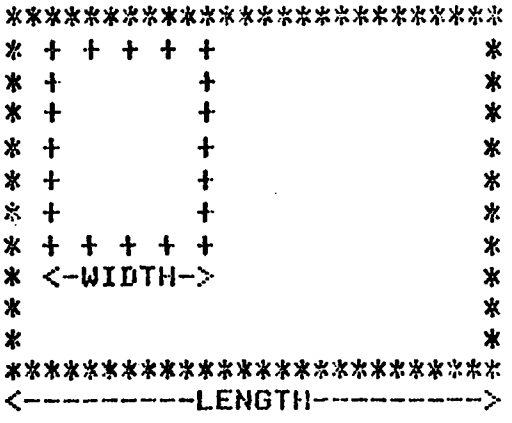
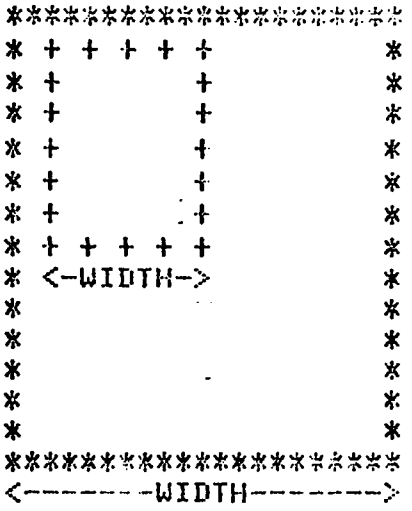
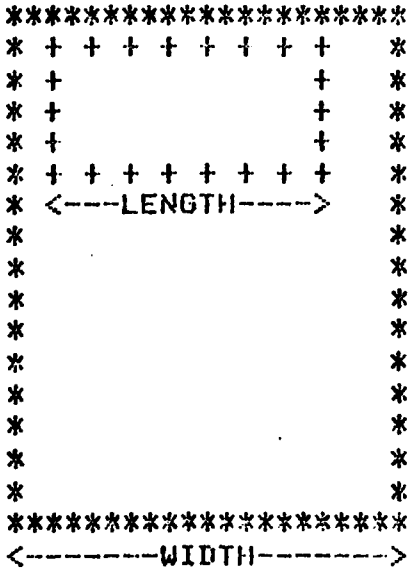

BOX 1
BOX 2
BOX 3
BOX 4
<--UNEVEN INITIAL LOAD
NEW
"STARTING SIDE"
PALLET

FIGURE 12


<-WIDTH->
<----------LENGTH---------->
ORIENTATION (A)


<-WIDTH->
<-------WIDTH------->
ORIENTATION (B)


<---LENGTH---->
<--------WIDTH-------->
ORIENTATION (C)

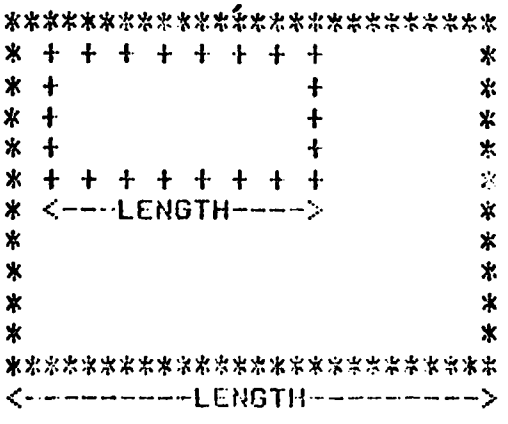

<---LENGTH---->
<----------LENGTH---------->
ORIENTATION (D)

FIGURE 13

IN THIS SOLUTION.

NOW LET US LOOK AT THE ACTUAL COMPUTER CODES.

EACH OF THE FIVE HEURISTICS ARE CONTAINED IN THE FIVE MODULES(OR SUBROUTINES) OF THE MAIN PROGRAM, "PALLET LOADING ROUTINE (PLR)". THESE FIVE MODULES CONSISTS OF SEVERAL BASIC SUBROUTINES, ARRANGED SO THAT THE PROPER LOADING ALGORITHM IS ACHIEVED. THE FOLLOWING IS A LIST OF THE FIVE MODULES, THE SUBROUTINES CONTAINED IN EACH AND THE CLASS IT BELONGS TOO.

MODULE NAME	SUBROUTINES USED	CLASS
MAST2A	LINFIT,NEWLST,PILE2A,SORT	1
MAST2B	LINFIT,NEWLST,PILE2B,SORT	1
MAST2C	LINFIT,NEWLST,PILE2A,SORT	1
MAST2D	LINFIT,NEWLST,PILE2B,SORT	1
MAST3A	LINFIT,NEWLST,REORP,PILE3&4,SORT	2

EACH OF THE MODULES ARE CALLED WITH THE NUMBER OF BOXES AVAILABLE FOR LOADING, 'N', AND AN 'AREA' VARIABLE WHICH IS USED TO RETURN THE TOTAL AREA LOADED BY EACH MODULE.

THE FOLLOWING SECTIONS DESCRIBE IN DETAIL THE OPERATION OF THE MAIN PROGRAM, THE MODULES AND THEIR SUBROUTINES. THE APPENDIX CONTAINS A LISTING OF ALL THE COMPUTER CODES AND ARE REFERED TO BY NAME AND LINE NUMBER IN THE FOLLOWING DISCUSSIONS.

[A] PALLET LOADING ROUTINE (PLR)

PLR IS THE MAIN DRIVER OF THE ENTIRE PROGRAM. IT CONTROLS WHICH OF THE FIVE MODULES ARE TO BE EXECUTED AND WHEN. IT ALSO KEEPS TRACK OF THE MODULE THAT PRODUCES THE LARGEST LOADED AREA. THE FOLLOWING IS A LINE-BY-LINE DESCRIPTION OF PLR.

LINE(1): ALL VARIABLES ARE TO BE CONSIDERED INTEGER. THIS IS BECAUSE THROUGH OUT THE PROGRAM THE ONLY OPERATIONS PERFORMED ON ANY VARAIBLE WILL BE ADDITION OR SUBTRACTION. THUS REAL OR FLOATING POINT IS NOT NECESSARY, AND WILL SAVE COMPUTATION TIME.

LINE(2): THE DATA SET WHICH CONTAINS THE LENGTHS AND WIDTHS OF THE BOXES TO BE LOADED, CALLED 'BANS4.DAT', IS ASSIGNED LOGICAL UNIT NUMBER 1.

LINE(3): LOGICAL UNIT 5 IS ASSIGNED TO 'TI:' OR THE TERMINAL BEING USED BY THE USER OF THE PROGRAM. THE OUTPUT OF THE PROGRAM WILL BE DISPLAYED ON THIS DEVICE. THIS ASSIGNMENT CAN BE CHANGED TO INDICATE ANY OUTPUT DEVICE REQUIRED BY THE USER.

LINE(4): DEFINE FILE 1 ASSOCIATES LOGICAL UNIT 1(I.E. THE UNFORMATTED, DIRECT ACCESS FILE, I.E. BAM34.DAT.

LINE(5): INITIALIZES 'NAREA' TO ZERO. THE VARIABLE 'NAREA' WILL EVENTUALLY CONTAIN THE LARGEST AREA LOADED BY ANY MODULE.

LINES(6) THRU (17): CONTAIN THE CONTROLING DO LOOP OF 'PLR'.

LINE(6): IS THE DO STATEMENT. AS THE INDEX, J, IS INCREMENTED FROM 1 TO 5 ALL STATEMENTS FOLLOWING THE "DO" WILL BE EXECUTED 5 TIMES. THIS ALLOWS THE PROGRAM TO RUN THROUGH ALL FIVE MODULES.

LINE(7): READS IN FROM LOGICAL UNIT 1 THE NUMBER OF BOXES TO BE LOADED AND GIVES VARIABLE 'N' THIS VALUE.

LINE(8): INITIALIZES 'AREA' TO ZERO. THIS IS REQUIRED BEFORE EACH MODULE IS RUN BECAUSE THE VALUE OF 'AREA' WILL BE CHANGED DEPENDING ON THE LOADING ROUTINE USED. 'AREA' IS USED TO RETURN THE AREA LOADED BY EACH MODULE.

LINES(9) THRU (13): THESE STATEMENTS CALL THE APPROPRIATE MODULE DEPENDING ON THE CURRENT VALUE OF THE DO STATEMENT INDEX J. THE VALUE OF 'N' AND 'AREA' ARE THE ARGUMENT OF EACH MODULE AND HAVE THE INITIAL VALUES ASSIGNED IN LINES(7) AND (8).

LINES(14) THRU (16): IN THESE LINES THE AREA LOADED BY EACH MODULE 'AREA' IS COMPARED TO THE CURRENT VALUE OF 'NAREA'. IF 'AREA' IS LARGER THAN 'NAREA' IT IS SAVED BY ASSIGNING ITS VALUE TO 'NAREA'. ALSO THE NUMBER OF THE MODULE THAT PRODUCED THIS LARGER AREA WILL BE SAVED AS 'OPT'. IF 'AREA' IS SMALLER THAN 'NAREA' THE CURRENT VALUE OF 'NAREA' AND 'OPT' IS RETAINED.

LINE(17): IF 'J' IS LESS THAN 5 RETURN TO LINE (6) AND REPEAT ALL THE STATEMENTS IN THE DO LOOP. IF 'J' IS EQUAL TO 5 PROCEED TO THE NEXT LINE.

LINE(18) AND (19): WHEN THE DO LOOP IS COMPLETED, THAT IS WHEN IT HAS PERFORMED ALL 5 MODULES LINES(18) AND (19) WILL WRITE THE VALUES OF 'NAREA', THE LARGEST LOADED AREA AND 'OPT', THE COORESPONDING OPTION TO THE OUTPUT DEVICE SPECIFIED IN LINE(3).

THE NEXT SET OF CODES TO BE EXAMINED WILL BE THE FIVE

MODULES THAT ARE CALLED IN 'PLR'. REMEMBER, THERE ARE TWO CLASSES OF MODULES, CLASS 1 AND CLASS 2. SINCE ALL FOUR CLASS 1 MODULES FOLLOW THE SAME PROCEDURE, WITH ONLY CHANGES IN THE PALLET-BOX ORIENTATION, ONLY MODULE 'MAST2A' WILL BE DESCRIBED IN DETAIL. DISCUSSION OF THE REMAINING THREE CLASS 1 MODULES WILL CONSIST OF POINTING OUT THE DIFFERENCES RELATED TO EACH PARTICULAR PALLET-BOX ORIENTATION.

MODULE 5, 'MAST3A', WILL THEN BE DISCUSSED IN DETAIL.

[B] MODULE MAST2A

MODULE 'MAST2A' IS CALLED UPON TO LOAD 'N' BOXES ONTO A PALLET WITH LENGTH = 104 INCHES AND WIDTH = 84 INCHES.

LINE(1): THIS IS THE MODULE NAME AND THE ARGUMENTS 'N' AND 'AREA'. WHEN 'MAST2A' IS CALLED BY 'PLR', 'N' AND 'AREA' ARE ASSIGNED THE VALUES THEY WERE GIVEN IN 'PLR'.

LINE(2): ESTABLISHES ALL VARIABLES AS INTEGER.

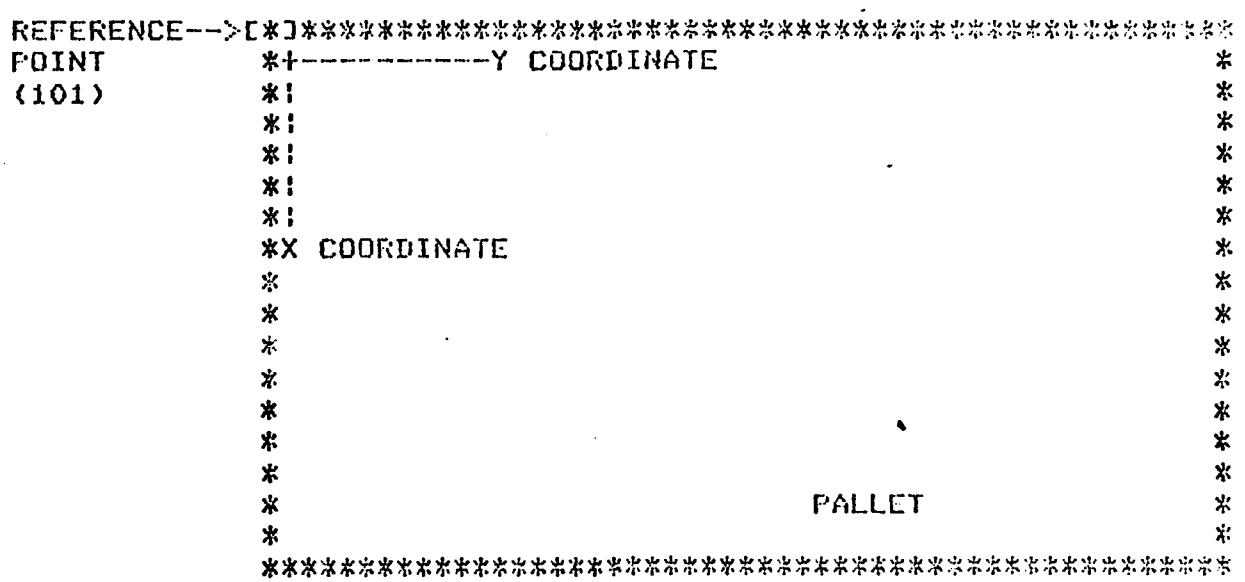
LINE(3): THIS DIMENSION STATEMENT DECLARES THE NAMED ARRAYS TO BE A MAXIMUM OF 30 LOCATIONS. REMEMBER, NO MORE THAN 30 BOXES CAN BE LOADED AT ONE TIME. THE ARRAYS ARE GENERALLY DIVIDED INTO SETS OF TWO. FOR EXAMPLE, 'BOXLEN(*) AND BOXWTH(*) REFER TO THE LENGTH AND WIDTH OF ONE OF THE BOXES CONTAINED IN THOSE TWO ARRAYS. THE EXCEPTION TO THIS, IS THE ARRAY 'COOR' WHICH WILL CONTAIN THE COORDINATES OF EACH BOX THAT IS LOADED.

LINE(4): THE DATA STATEMENT INITIALIZES SEVERAL SINGLE VARIABLES AND ARRAYS TO ZERO.

LINE(5): 'POINT' IS INITIALIZED WITH THE VALUE 101. THIS INITIAL VALUE OF POINT IS THE COORDINATE OF THE PALLET 'REFERENCE POINT'. THIS 'REFERENCE POINT' IS ALWAYS LOCATED IN THE UPPER LEFT HAND CORNER OF THE PALLET. ALSO, THIS VARIABLE IS USED TO KEEP TRACK OF THE POSITION OF EACH OF THE BOXES AS THEY ARE LOADED ON THE PALLET. THE 'REFERENCE POINT' VALUE, 101, IS BASED ON THE COORDINATE SYSTEM SHOWN IN FIGURE 14. THE LAST TWO NUMBERS OF 'POINT' IS THE Y-COORDINATE MEASURED, IN INCHES, DOWN FROM THE 'REFERENCE POINT'. THE FIRST THREE NUMBERS IN 'POINT' IS THE THE X-COORDINATE MEASURED, IN INCHES, TO THE RIGHT OF THE REFERENCE POINT. FOR EXAMPLE, THE PALLET 'REFERENCE POINT' IS 101, SO THIS POINT IS EXACTLY THE UPPER LEFT HAND CORNER OF THE PALLET.

LINE(6) AND (7): THIS DO LOOP WILL READ INTO ARRAYS 'BL' AND 'BW' THE LENGTHS AND WIDTHS OF THE 'N' BOXES.

LINE(8): THE SUBROUTINE 'SORT' IS CALLED. THE ARGUMENTS ARE THE ARRAYS 'PTLOK', WITH ALL ELEMENTS EQUAL TO ZERO, AND THE NUMBER 30.



PALLET COORDINATE SYSTEM

FIGURE 14

WHEN THE SUBROUTINE RETURNS, THE ARRAY 'BL' WILL BE UNCHANGED. BUT 'PTLGH' WILL CONTAIN A SET OF POINTERS THAT WILL ALLOW SORTING OF THE ARRAY 'BL' IN DESCENDING VALUES OF LENGTH. 'STLGH' WILL HAVE THE VALUE OF THE BOX LENGTH IN 'BL' FROM WHICH THE ARRAY 'PTLGH' WILL START ITS SORT.

LINE(9): SETS 'M' EQUAL TO 'STLGH'.

LINE(10) THRU (15): THIS SECOND DO LOOP NOW MAKES USE OF THE POINTER ARRAY 'PTLGH' TO SORT 'BL', IN THE FOLLOWING MANNER;

STARTING WITH THE BOX LENGTH IDENTIFIED BY 'BL(M)' THIS VALUE IS COPIED INTO 'BOXLEN(K)' AND 'MBL(K)', WHERE 'K' IS THE INDEX OF THE DO LOOP. ALSO, 'BW(M)' IS COPIED INTO 'BOXWTH(K)' AND 'MBW(K)'.

NEXT, THE VALUE OF 'M' IS CHANGED TO THE POINTER VALUE ASSOCIATED WITH 'BL(M)'. 'PTLGH(M)' IS THEN USED TO FIND THE NEXT SMALLEST BOX LENGTH IN 'BL'. THE DO LOOP THEN RETURNS TO LINE(10) AND REPEATS THE OPERATION. THE DO LOOP FINISHS WHEN ALL 'N' BOXES HAVE BEEN SORTED.

LINE(16): 'A' IS GIVEN THE VALUE OF 'N'. THIS IS DONE BECAUSE 'N' WILL BE CHANGED LATER IN THE PROGRAM.

LINE(17): 'SMALL' IS SET EQUAL TO 4, WHICH IS THE VALUE OF THE SMALLEST BOX LENGTH FROM THE TOTAL DATA SET OF BOXES. THIS VALUE WILL BE USED THROUGHOUT THE PROGRAM AS THE DIFFERENTIAL VALUE USED IN MAKING A UNIFORM LOAD ASSUMPTION (SEE PARA 6, UNIFORM LOAD ASSUMPTION).

LINE(18): SUBROUTINE 'NEWLST' IS CALLED. SEE SECTION [E] FOR EXPLAINATION OF THE ARGUMENTS FOR THIS SUBROUTINE. THE ARGUMENT VALUES SENT TO 'NEWLST' BY THIS CALL STATEMENT ARE;

N	= NUMBER OF BOXES TO BE LOADED
104	= PALLET LENGTH
84	= PALLET WIDTH
SMALL	= 4
SPACE	= TRACK = WIDE = N1 = STPT = 0
A	= N
BOXLEN	= MBL = ARRAYS OF 'N' BOX LENGTHS
BOXWTH	= MBW = ARRAYS OF 'N' BOX WIDTHS
LBOX	= WBOX = TLBOX = TWBOX = COOR = ARRAYS WITH ALL 'N' ELEMENT EQUAL TO ZERO
POINT	= 101

'NEWLST' PERFORMS THE LOADING OPERATION DESCRIBED IN THE FIRST PART OF PARA 5. THAT IS, 'NEWLST' WILL PERFORM THE INITIAL LOAD AND AS MANY LOADS THEREAFTER IN ACCORDANCE WITH STEPS(1) THRU (5) AND STEP (6'), UNTIL AN "UNEVEN" LOAD IS OBTAINED.

LINE(19): AFTER RETURNING FROM 'NEWLST' A TEST IS MADE TO SEE IF ALL THE BOXES HAVE BEEN LOADED. IF THERE ARE NO MORE BOXES TO LOAD THE RETURNING VALUE OF 'N' WILL BE ZERO AND THE PROGRAM

WILL PROCEED TO LINE(22), WHICH BEGINS THE OUTPUT PROCESS. IF 'N' IS NOT ZERO THEN THE PROGRAM EXECUTES THE NEXT LINE.

LINE(21): HERE THE SUBROUTINE 'PILE2A' IS CALLED TO BEGIN THE "STACKING" PORTION OF THE PROGRAM. THE ARGUMENTS CONTAIN THE FOLLOWING VALUES:

BOXLEN	= AN ARRAY OF 'A' ELEMENTS EACH WITH A VALUE OF A BOX LENGTH. HOWEVER, THOSE BOXES THAT HAVE BEEN LOADED BY SUBROUTINE 'NEWLST' OR CAN NOT FIT IN THE REMAINING SPACE ON THE PALLET, WILL HAVE NEGATIVE VALUES.
BOXWTH	= AN ARRAY OF 'A' ELEMENTS EACH WITH A VALUE OF A BOX WIDTH.
104	= PALLET LENGTH.
N	= NUMBER OF BOXES NOT LOADED BY 'NEWLST'.
LBOX	= ARRAY OF (A-N) ELEMENTS EACH HAVING THE VALUE OF A BOX LENGTH OF THOSES BOXES LOADED BY 'NEWLST'.
WBOX	= ARRAY OF (A-N) ELEMENTS WITH THE BOX WIDTHS OF THOSE BOXES LOADED BY 'NEWLST'.
TRACK	= A-N = NUMBER OF BOXES LOADED BY 'NEWLST'.
SMALL	= 4
WIDE	= LARGEST PALLET WIDTH REMAINING AFTER BOXES WHERE LOADED BY 'NEWLST'. THIS DOES NOT INCLUDE THE "LAST" ROW OF BOXES, IF MORE THAN ONE LOAD IS IDENTIFIED BY 'NEWLST'.
MBL	= ARRAY OF 'N' BOX LENGTHS, ALL POSITIVE IN VALUE.
MBW	= ARRAY OF 'A' BOX WIDTHS, ALL POSITIVE IN VALUE.
COOR	= ARRAY OF 'A' ELEMENTS. THE ENTRIES IN THIS ARRAY COORESPONDING TO THE BOXES LOADED BY 'NEWLST' WILL HAVE A 5 DIGIT NUMBER REPRESENTING THE COORDINATE OF THE UPPER LEFT CORNER OF THE BOX. ALL OTHER ELEMENTS WILL REMAIN ZERO.
POINT	= COORDINATE OF THE LAST BOX TO BE LOADED BY 'NEWLST'.
A	= ORIGINAL NUMBER OF BOXES; A = TRACK+N.

SUBROUTINE PILE2A PERFORMS THE "STACKING" PROCEDURE DESCRIBED IN PARA 5, STEPS (6)(B), (7) AND (8), USING ONLY THOSE BOXES NOT LOADED BY 'NEWLST'.

'PILE2A' LOADS BOXES WITH THE BOX LENGTH PARALLEL TO THE PALLET 'STARTING SIDE'. LATER 'PILE2B' WILL BE USED TO LOAD THE BOXES PERPENDICULAR TO THE 'STARTING SIDE'.

LINE(24) THRU (27): IN THIS DO LOOP THE PROGRAM OUTPUTS THREE LISTS, EACH WITH 'N' ELEMENTS. THESE LISTS WILL CONTAIN THE BOX LENGTHS, WIDTHS, AND COORDINATES. A SAMPLE OUTPUT IS SHOWN IN FIGURE 14. THE LENGTH OF THOSE BOXES THAT HAVE BEEN LOADED ARE SHOWN AS NEGATIVE VALUES.

LINE(28) THRU (33): THIS FINAL DO LOOP COMPUTES THE VALUE OF 'AREA' LOADED BY MAST2A AND THE TOTAL NUMBER OF BOXES USED.

LINE(34) AND (35): OUTPUTS THE RESULTS COMPUTED IN LINES (28) TO (34).

LINE(36): THE COMPUTER RETURNS TO THE MAIN PROGRAM, 'PLR', WITH;

N = NUMBER OF BOXES LOADED
AREA = TOTAL AREA COVERED BY THE LOADED BOXES.

[C] MODULES MAST2B, MAST2C, MAST2D

EACH OF THESE THREE MODULES EXACTLY DUPLICATES THE PROCESS DESCRIBED IN [B]. HOWEVER, IN EACH OF THESE MODULES THE PALLET-BOX ORIENTATION IS DIFFERENT. BELOW IS A LIST OF CHANGES, BY MODULE, THAT COORESPONDS TO THE CHANGE IN PALLET-BOX ORIENTATION. REMEMBER, 'MAST2A' HAD A PALLET LENGTH OF 104 INCHES AND WIDTH OF 84 INCHES, WITH THE BOX LENGTHS PARALLEL TO THE PALLET LENGTH.

MAST2B

THE ONLY CHANGE IN THIS MODULE IS THE USE OF SUBROUTINE 'PILE2B', WHICH WHEN CALLED UPON TO 'STACK' BOXES WILL ORIENTATE THE BOX WIDTHS PARALLEL TO THE PALLET LENGTH.

MAST2C

IN THIS MODULE THE PALLET IS ROTATED 90 DEGREES SO THAT ITS LENGTH IS NOW 84 INCHES AND ITS WIDTH IS 104 INCHES. 'PILE2A' IS USED IN THE 'STACKING' PROCEDURE, WHICH PUTS THE BOX LENGTHS PARALLEL TO THE PALLET WIDTH.

MAST2D

FINALLY, THIS MODULE WILL USE A PALLET WIDTH OF 84 INCHES AND ORIENTATE THE BOX WIDTHS PARALLEL TO THE PALLET WIDTH USING SUBROUTINE 'PILE2B'.

[D] MODULE MAST3A

MODULE 'MAST3A' REMEMBER IS A CLASS 2 ROUTINE. HOWEVER 'MAST3A' IS VERY SIMILAR TO 'MAST2A'. LINES (1) TO (19) AND LINES (22) TO (37) IN 'MAST3A' HAVE EXACTLY THE SAME INTERPRETATION AS THOSE LINES IN 'MAST2A'. THE MAJOR DIFFERENCE IN THE TWO MODULES IS IN LINE (21). IN 'MAST3A' SUBROUTINE 'PILE4' IS USED INSTEAD OF 'PILE2A'. WITH 'PILE4' THE PROCESS DESCRIBED IN PARA 5, STEPS (6)(B), (7), AND (8)

IS IMPLEMENTED. THIS DIFFERENCE IN LOADING PROCEDURE WILL RESULT IN A VERY DIFFERENT FINAL LOADING PATTERN.

A DETAILED EXPLANATION OF THE SUBROUTINE 'PILE4' AND IS EFFECT ON THE LOADING PROCEDURE CAN BE FOUND IN SECTION [F].

NOW, LETS EXAMINE THE MANY SUBROUTINES THAT MAKE UP THE FIVE DIFFERENT MODULES. THE FIRST OF THESE SUBROUTINES IS 'NEWLST'.

[E] SUBROUTINE NEWLST

'NEWLST' IS THE LONGEST OF THE SUBROUTINES AND IS COMMON TO ALL FIVE MODULES. 'NEWLST' IS THE SUBROUTINE THAT PERFORMS THE INITIAL LOADING ROUTINE DESCRIBED IN PARA 5, STEPS (1) THRU (5), REPEATING AS NECESSARY IN ACCORDANCE WITH THE 'UNIFORM LOAD' ASSUMPTION (SEE PARA 6).

LINE(1): THIS IS THE SUBROUTINE NAME AND A LIST OF ITS ARGUMENTS. THE VALUE OF THESE ARGUMENTS WERE DETERMINED IN THE MODULE THAT CALLED 'NEWLST'.

LINE(2): ESTABLISHES ALL VARIABLES AS INTEGERS.

LINE(3): DIMENSION STATEMENT DECLARES ALL ARRAYS TO HAVE A MAXIMUM OF 30 ELEMENTS. THE USE OF EACH ARRAY WILL BECOME EVIDENT LATER IN THE SUBROUTINE.

LINE(4): GIVES 'NEWWTH' THE VALUE OF 'WIDTH', THE WIDTH OF THE PALLET.

LINE(5) THRU (12): THE FIRST DO STATEMENT EXAMINES ALL 'N' BOXES TO DETERMINE WHICH ONES HAVE BEEN LOADED.

FIRST, THE 'BOXLEN' VALUE OF THE BOXES IS TESTED TO SEE IF IT IS NEGATIVE. THAT IS, HAS THE BOX ALREADY BEEN LOADED. IF THE 'BOXLEN' VALUE IS NEGATIVE THEN THE REMAINING PORTION OF THE LOOP IS DISREGARDED AND THE NEXT BOX IS EXAMINED. BUT IF THE 'BOXLEN' VALUE IS POSITIVE THE 'BOXWTH' VALUE IS TESTED TO SEE IF THE BOX WILL FIT IN THE REMAINING PALLET WIDTH, 'NEWWTH'. IF THE 'BOXWTH' WILL FIT, THE BOX IS DISREGARDED AND THE LOOP PROCEDES TO TEST THE NEXT BOX. IF HOWEVER, THE 'BOXWTH' IS TOO LARGE THE LOOP CONTINUES TO THE NEXT STATEMENT. THUS, IF THE 'BOXLEN' VALUE IS NOT NEGATIVE (MEANING IT IS AVAILABLE FOR LOADING) BUT THE 'BOXWTH' IS LARGER THAN 'NEWWTH' BOTH THE 'BOXLEN' AND 'BOXWTH' VALUES ARE MADE NEGATIVE, INDICATING IT CAN NOT BE USED IN THE LOADING ROUTINE.

LINE(13): 'SPACE' IS GIVEN THE VALUE OF ZERO. 'SPACE' WILL BE USED WHEN THE INITIAL LOAD IS PERFORMED (SEE LINE(14)).

LINE(14): SUBROUTINE 'LINFIT' IS CALLED. 'LINFIT' SEEKS TO FIT AS MANY BOXES, FROM A LIST OF 'N' BOX LENGTHS, 'BOXLEN', INTO A LINEAR SPACE OF SIZE 'LENGTH'. ONLY THOSE BOXES WITH POSITIVE 'BOXLEN' VALUES WILL BE CONSIDERED. THE OPTIMAL

COMBINATION OF BOXES, THAT MINIMIZES THE SLACK SPACE 'SPACE', IS CHOOSEN. THE BOXES IDENTIFIED IN THIS OPTIMAL COMBINATION WILL HAVE THEIR 'BOXLEN' VALUES MADE NEGATIVE.

LINES (15), (16) AND (17): INITIALIZES THE VARIABLES 'COUNT', 'TRACK' AND 'NUM' TO ZERO.

LINES(18) THRU (35): IN THIS DO LOOP ALL 'N' BOXES ARE TESTED. EACH BOX WILL FALL INTO ONE OF THE FOLLOWING CATEGORIES:

(A) BOX HAS ALREADY BEEN LOADED, OR WILL NOT FIT IN AN AREA OF DIMENSIONS 'LENGTH' BY 'NEWWTH'.
(B) THE BOX HAS BEEN IDENTIFIED BY SUBROUTINE 'LINFIT' AS PART OF THE OPTIMAL INITIAL LOAD.
(C) THE BOX IS STILL AVAILABLE FOR USE.

WHEN THE BOXES ARE TESTED, IF:

(AA) THE BOX LENGTH AND BOX WIDTH ARE BOTH NEGATIVE THE THE BOX FALLS IN CATAGORY (A).
(BB) THE BOX LENGTH IS NEGATIVE AND THE BOX WIDTH IS POSITIVE THE BOX BELONGS TO CATAGORY (B).
(CC) BOTH BOX LENGTH AND WIDTH ARE POSITIVE A CATAGORY (C) BOX IS IDENTIFIED.

IN THE CASE OF CATEGORY (B) BOXES, THE ABSOLUTE VALUE OF THE BOX LENGTH AND WIDTH ARE STORED IN ARRAYS 'LBOX', 'TLBOX' AND 'WBOX', 'TWBOX' RESPECTIVELY; LINES (23), (24), AND (27), (28). THE NUMBER OF BOXES IS COUNTED BY INCREMENTING 'TRACK' AND 'N1', LINES (23) AND (26).

WHEN A CATEGORY (C) BOX IS FOUND THE RESPECTIVE BOX LENGTH AND WIDTH IS STORED IN 'LLBOX' AND 'WWBOX'. THE TALLY OF THESE BOXES IS KEPT BY INCREMENTING 'NUM'.

LINE(36): 'N' IS GIVEN THE VALUE OF 'NUM', THE NUMBER OF BOXES THAT REMAIN TO BE LOADED.

LINE(37): IF NO BOXES WERE IDENTIFIED FOR THE INITIAL LOAD BY 'LINFIT', THE PROGRAM RETURNS TO THE CALLING MODULE.

LINE(39): IF THE NUMBER OF BOXES REMAINING AVAILABLE FOR LOADING IS ZERO, THEN THE PROGRAM WILL RETURN TO THE CALLING MODULE.

LINE(41): INITIALIZES 'NEWLEN' TO THE LENGTH OF THE FIRST BOX IDENTIFIED BY 'LINFIT'.

LINE(42): 'WIDE' IS GIVEN THE VALUE OF 'NEWWTH'.

LINE(43): A TEST IS MADE TO SEE IF 'LINFIT' IDENTIFIED ONLY ONE BOX. IF THIS IS THE CASE THE PROGRAM PROCEEDS TO LINE (58); OTHERWISE THE PROGRAM GOES TO THE NEXT STATEMENT.

LINES (47) THRU (55): IN THIS DO LOOP ALL THE BOXES IDENTIFIED BY 'LINFIT' ARE EXAMINED. EACH BOX'S WIDTH IS COMPARED BY MEANS OF FIRST, A TWO-BOX COMPARISION, THEN A THREE-BOX COMPARISION AND SO ON UNTIL ALL BOX WIDTHS HAVE BEEN COMPARED TO EACH OTHER. AFTER EACH COMPARISION THE MAXIMUM DIFFERENCE IN BOX WIDTHS IS RECORDED IN 'SDIFF' AND THE LARGEST BOX WIDTH IS RECORDED IN 'DIFF'.

ALSO, AS EACH COMPARISION TAKES PLACE 'NEWLEN' IS UPDATED. THAT IS, 'NEWLEN', WHICH WAS ORIGNALLY EQUAL TO THE LENGTH OF THE FIRST BOX IDENTIFIED BY 'LINFIT', IS INCREASED BY THE APPROPRIATE BOX LENGTH.

THE VALUE OF 'NEWWTH', OR THE REMAINING SPACE ALONG THE SIDE OF THE PALLET, IS ALSO CHANGED BY SUBTRACTING THE APPROPRIATE BOX WIDTH.

WHEN THE PROGRAM EXITS THE DO LOOP THE VALUES OF 'NEWLEN' AND 'NEWWTH' CAN BE INTERPRETED AS;

NEWLEN = THE SUM OF ALL ENTRIES OF ARRAY 'LBOX'.
= THE SUM OF THE LENGTHS OF EACH BOX IDENTIFIED BY 'LINFIT'.
NEWWTH = THE PREVIOUS 'NEWWTH' VALUE MINUS THE MAXIMUM WIDTH FROM ALL THE BOXES IDENTIFIED BY 'LINFIT'.

THIS INFORMATION WILL BE USED LATER IN CHECKING TO SEE IF THE "UNIFORM LOAD ASSUMPTION" IS VALID.

LINE(56): 'WIDE' IS GIVEN THE NEW VALUE EQUAL TO 'NEWWTH'.

LINE(57): THE PROGRAM SKIPS TO LINE (61).

LINES(58) THRU (60): THE PROGRAM WILL SKIP TO THIS LINE, DISREGARDING LINES (46) THRU (57) IF THE CONDITION IN LINE (45) IS TRUE, I.E. ONLY ONE BOX IS IDENTIFIED BY 'LINFIT'.
HERE, 'NEWLEN' IS MADE EQUAL TO 'LENGTH', AND THE WIDTH OF THE BOX IS SUBTRACTED FROM 'NEWWTH'. ALSO, 'WIDE' IS GIVEN THE NEW VALUE OF 'NEWWTH'.

LINE(61) THRU (63): THIS DO LOOP SIMPLY CHANGES ARRAYS 'BOXLEN' AND 'BOXWTH' TO CONTAIN ONLY THOSE BOXES THAT WERE IDENTIFIED EARLIER AS BEING AVAILABLE FOR FUTURE LOADING.

LINE(64): 'TPT' IS GIVEN THE VALUE OF 'POINT'.

LINES(65) THRU (76): TWO NESTED DO LOOPS ARE USED TO ASSIGN A X AND Y COORDINATE TO EACH BOX IDENTIFIED BY 'LINFIT'. THESE COORDINATES ARE PLACED IN THE APPROPRIATE ENTRY OF THE ARRAY 'COOR'. THIS IN EFFECT "LOADS" THE BOXES ONTO THE PALLET.

LINES(77) THRU (80): HERE THE VALUE OF 'POINT' IS CHANGED TO COORESPOND TO A POSITION ALONG THE LEFT SIDE OF THE PALLET AND A VALUE OF 'WIDTH' MINUS 'NEWWTH' BELOW THE "REFERENCE POINT".

LINE(81): IF THE MAXIMUM DIFFERENCE OF ANY TWO BOXES IDENTIFIED BY 'LINFIT' IS GREATER THAN 4 INCHES AN "UNEVEN LOAD" EXISTS AND THE PROGRAM RETURNS TO THE CALLING MODULE. IF THE DIFFERENCE IS LESS THAN 4 INCHES THE PROGRAM EXECUTES THE NEXT STATEMENT.

LINE(83): IF THE DIFFERENCE BETWEEN 'LENGTH' AND THE SUM OF THE 'LINFIT' BOX LENGTHS IS GREATER THAN 'SMALL'; OR 'NEWWTH', THE SPACE REMAINING BELOW THE 'LINFIT' BOXES, IS LESS THAN 'SMALL', THE PROGRAM WILL RETURN. IF NEITHER CONDITION IS TRUE THE NEXT STATEMENT IS EXECUTED.

LINE(85): 'NEWLEN' IS INITIALIZED TO THE VALUE OF 'LENGTH'.

LINE(86): BECAUSE THE CONDITIONS IN LINES (81) TO (84) WERE SATISFIED, A "UNIFORM LOAD" CAN BE ASSUMED, AND THE PROGRAM RETURNS TO LINE(5) TO BEGIN THE SECOND LOAD USING THE SAME PROCEDURE BUT WITH AN UPDATED LIST OF BOXES AND NEW VALUES FOR 'NEWLEN' AND 'NEWWTH'.

LINE(87): RETURN TO CALLING MODULE.

LINE(88): END OF SUBROUTINE.

[F] SUBROUTINE PILE2A

THIS SUBROUTINE PERFORMS THE "STACKING" SEQUENCE OF THE PROGRAM. BEGINNING WITH 'TRACK' NUMBER OF BOXES, HAVING LENGTHS AND WIDTHS CONTAINED IN ARRAYS 'LBOX' AND 'WBOX', AS MANY ADDITIONAL BOXES FROM A LIST OF 'N' BOXES ARE LOADED BELOW EACH 'LBOX'.

LINE(1): THE NAME OF THE SUBROUTINE AND THE VARIOUS ARGUMENTS. SEE SECTION [B] FOR AN EXPLAINATION OF THE VALUE OF EACH ARGUMENT WHEN 'PILE2A' IS CALLED.

LINE(2): ALL VARIABLES ARE INTEGER.

LINE(3): DIMENSIONS ALL ARRAYS TO 30 ELEMENTS.

LINES(4) THRU (6): INITIALIZES 'SAVE', 'START' AND 'SWIDE' EQUAL TO ZERO.

LINE(7): 'BPT' IS GIVEN THE VALUE OF 'POINT'.

LINES(8) THRU (11): THE VARIABLE 'SWIDE' IS GIVEN A VALUE EQUAL TO 'WIDE' PLUS THE LARGEST WIDTH OF ALL BOXES IN 'WBOX'.

LINE(12): 'TT2' IS INITIALIZED AS ZERO.

LINE(13): THIS DO STATEMENT STARTS A DO LOOP THAT CONTAINS THE REMAINDER OF THE SUBROUTINE. NESTED IN THIS LARGE DO LOOP ARE SEVERAL SMALLER DO LOOPS THAT ARE USED TO "STACK" BOXES. THE

LARGER DO LOOP IS INCREMENTED BY ONE UNTIL STACKING HAS OCCURED BELOW EACH BOX IN 'LBOX'.

LINE(14): 'POINT' IS EQUAL TO 'BPT'. THIS CHANGE IS REQUIRED, AT THE BEGINNING OF THE LARGE DO LOOP TO KEEP TRACK OF WHAT THE COORDINATE OF THE FIRST BOX LOADED UNDER EACH SUCESSIVE BOX IN 'LBOX'.

LINE(15): 'TEMP' IS EQUAL TO ZERO.

LINE(16): IF THE LENGTH OF A PARTICULAR BOX, IN 'LBOX', WHICH IS TO BE LOADED UNDER, IS LESS THAN OR EQUAL TO 'START', THE PROGRAM SKIPS TO LINE (71) AND ON TO LINE (13) TO BEGIN LOADING UNDER THE NEXT BOX IN 'LBOX'.
THE VALUE OF 'START' WILL BECOME APPARENT IN LINE (29). 'START' IS, IN FACT, A NUMBER CORRESPONDING TO THE LAST BOX IN 'LBOX THAT HAS THE SAME WIDTH, WITHIN 4 INCHES, AS ALL THE PREVIOUS BOXES CONSIDERED.

LINE(18): 'NEWLEN' IS GIVEN THE VALUE OF THE LENGTH OF THE CURRENT BOX IN 'LBOX'.

LINE(19): 'NEWWTH' IS EQUAL TO 'SWIDE' MINUS THE WIDTH OF THE CURRENT BOX IN 'WBOX'.

LINE(20): IF THE CURRENT BOX IS ALSO THE LAST BOX IN 'LBOX' THEN THE PROGRAM SKIPS TO LINE(32).

LINE(21): 'IP1' IS MADE EQUAL TO I+1, WHICH IS THE NEXT BOX IN 'LBOX'.

LINES(22) THRU (30): THIS DO LOOP PERFORMS THE TEST FOR A "UNIFORM LOAD" BETWEEN ANY NUMBER OF CONSECUTIVE BOXES IN 'WBOX'.
IF AN "UNIFORM LOAD" CAN BE ASSUMED WITH SOME OF THE BOXES 'NEWLEN' IS INCREASED AND 'NEWWTH' IS DECREASED BY THE APPROPRIATE AMOUNTS.

LINE(31): IF THE DO LOOP IS COMPLETED 'NEWWTH' IS SET EQUAL TO 'SWIDE' MINUS THE MAXIMUM BOX WIDTH OF THOSE BOXES IDENTIFIED IN LINES (22) THRU (30) AS HAVING MET THE TEST FOR A "UNIFORM LOAD" ASSUMPTION.

LINE(32): 'SAVE' IS INCREMENTED BY 'NEWLEN'.

LINE(33): IF 'I' IS EQUAL TO 'TRACK' THEN 'NEWLEN' IS GIVEN THE VALUE OF 'LENGTH'.

LINES(35) THRU (37): THESE STATEMENTS INITIALIZES SEVERAL SINGLE VARIABLES TO ZERO.

LINE(38): THIS DO LOOP BEGINS THE PROCESS OF "STACKING" BOXES UNDER THOSE BOXES CONTAINED IN 'LBOX'.

LINE(39): EACH OF THE BOXES IN 'BOXLEN' IS CHECKED TO SEE IF IT IS AVAILABLE FOR LOADING OR IF IT WILL FIT IN THE REMAINING SPACE, WITH

DIMENSIONS 'NEWLEN' BY 'NEWWTH'.

LINE(41): IF THE CURRENT BOX CAN BE LOADED 'NEWWTH' IS DECREMENTED BY THE 'BOXWTH' VALUE OF THIS BOX.

LINE(42): IF THE REMAINING SPACE ALONG 'NEWLEN' UNDER WHICH 'PILE2A' IS TRYING TO STACK IS LESS THAN 4 INCHES, THEN SKIP TO LINE (49). OTHERWISE PROCEED TO THE NEXT STATEMENT IN THE PROGRAM.

LINE(44): IF THE LENGTH OF THE CURRENT BOX JUST LOADED, IS LESS THAN 'MAX', SKIP TO LINE (50).

LINE(46): IF THE BOX'S LENGTH IS NOT LESS THAN 'MAX' GIVE 'MAX' THE VALUE OF 'BOXLEN' FOR THE CURRENT BOX UNDER CONSIDERATION.

LINE(47): 'TEMPT' IS UPDATED TO THE VALUE OF 'POINT' PLUS THE VALUE OF 'MAX', THAT IS THE LENGTH OF THE LOADED BOX.

LINE(48): THIS "GO TO" STATEMENT ALLOWS THE PROGRAM TO SKIP LINE (49) WHICH IS ONLY EXECUTED IF THE CONDITION IN LINE (42) IS TRUE.

LINE(49): IF THIS STATEMENT IS EXECUTED, THAT IS NO MORE BOXES CAN BE LOADED, 'WMAX' IS INCREASED BY THE WIDTH OF THE BOX JUST LOADED.

LINES(50) THRU (58): THIS DO LOOP IS USED TO ASSIGN A X AND Y COORDINATE TO THE BOX JUST IDENTIFIED TO BE LOADED. THIS COORDINATE IS PLACED IN THE APPROPRIATE ENTRY OF THE ARRAY 'COOR'. THIS IN EFFECT "LOADS" THE BOX ONTO THE PALLET.

LINE(59): HERE THE VALUE OF 'POINT' OR THE COORDINATE OF THE NEXT BOX TO BE LOADED, IS CHANGED TO CORRESPOND TO A POSITION DIRECTLY BELOW THE BOX JUST LOADED.

LINE(60): THE VALUE OF 'BOXLEN' FOR THE BOX JUST LOADED IS CHANGED TO A NEGATIVE. IN THIS WAY THIS BOX IS IDENTIFIED AS NOT BEING AVAILABLE FOR FUTURE LOADING CONSIDERATION.

LINE(61): THIS STATEMENT SEND THE PROGRAM BACK TO LINE (38) WHERE THE NEXT BOX IS CONSIDERED FOR "STACKING" IN THE AREA 'NEWLEN' BY 'NEWWTH'.

LINE(62): IF 'MAX' IS EQUAL TO ZERO AT THE END OF THE DO LOOP (LINES (38) THRU (61)) THE NUMBER OF BOXES "STACKED" WILL BE ZERO. IN THIS CASE THE PROGRAM GOES TO LINE (69).

LINES(64) THRU (67): IF 'MAX' IS NOT EQUAL TO ZERO THEN THE VALUES OF 'POINT', 'NEWLEN' AND 'TEMP' ARE UPDATED. THIS PREPARES THE PROGRAM FO STARTING TO STACK BOXES ALONG SIDE THE BOX OR BOXES JUST LOADED IN LINE (38) THRU (61).

LINE(68): RETURNS THE PROGRAM TO LINE(35) TO CONTINUE "STACKING".

LINE(69): IF THIS STATEMENT IS EXECUTED, THAT IS THE CONDITION IN LINE(62) IS TRUE, THE VALUE OF 'BPT' IS UPDATED TO A POSITION JUST BELOW THE NEXT GROUP OF BOXES IN 'LBOX' THAT THE SUBROUTINE

WILL BEGIN "STACKING" UNDER.

LINE(71): THE PROGRAM NOW RETURNS TO LINE (13) AND BEGINS THE "STACKING" PROCESS AGAIN UNDER THE NEW BOX(ES).

LINE(72): RETURNS TO CALLING MODULE.

LINE(73): END OF SUBROUTINE.

[G] SUBROUTINE PILE2B

THIS SUBROUTINE IS ALMOST IDENTICAL TO 'PILE2A'. IN BOTH SUBROUTINES BOXES ARE "STACKED" UNDER THOSE BOXES LOADED BY 'NEWLST' AND CONTAINED IN ARRAYS 'LBOX' AND 'WBOX'. HOWEVER, 'PILE2B' LOADS ALL THESE ADDITIONAL BOXES WITH THEIR WIDTHS PARALLEL TO THE "STARTING SIDE" OF THE PALLET RATHER THAN THEIR LENGTHS AS WAS THE CASE IN 'PILE2A'.

THIS CHANGE IS ACCOMPLISHED BY INTERCHANGING 'BOXLEN' AND 'BOWWTH' IN LINES (39) THRU (46) AND IN LINES (49) AND (59). THIS WILL PRESENT A COMPLETELY DIFFERENT LOADING PATTERN THAN DOES 'PILE2A'.

[H] SUBROUTINE PILE4

RECALL IN SECTION [D] THE MODULE MAST3A WAS SAID TO BE BASICALLY THE SAME AS MODULE MAST2A EXCEPT THAT 'PILE4' WILL START WITH A NUMBER OF BOXES, WITH LENGTH 'BOXLEN' AND WIDTH 'BOXWTH' WHICH REMAIN TO BE LOADED. THEN, UNDER EACH BOX IN THE "LAST" LOAD IDENTIFIED IN 'NEWLST', IT WILL LOAD AS MANY BOXES AS POSSIBLE WITH THE HELP OF SUBROUTINES 'NEWLST' AND 'PILE3' (SEE SECTION [I]). REMEMBER, 'NEWLST' WILL REPEATEDLY LOAD AS MANY BOXES AS POSSIBLE ALONG THE PALLET'S "STARTING SIDE" UNTIL AN "UNEVEN LOAD" APPEARS. 'PILE4' THEN IMPLEMENTS STEPS (6)(A),(7) AND (8) DESCRIBED IN PARA. 5.

LINE(1): THE SUBROUTINE NAME AND THE LIST OF ARGUMENTS. THE VALUE OF THE ARGUMENTS ARE ASSIGNED BY THE CALLING MODULE, MAST3A. THEY CAN BE INTERPRETED AS FOLLOWS:

- BOXLEN = AN ARRAY OF 'N' ELEMENTS EACH WITH A VALUE OF A BOX LENGTH. THOSE BOXES LOADED BY 'NEWLST' OR CAN NOT FIT IN THE REMAINING SPACE ON THE PALLET, WILL HAVE NEGATIVE VALUES.
- BOXWTH = AN ARRAY OF 'N' ELEMENTS EACH WITH A VALUE OF A BOX WIDTH.
- LENGTH = 104
- N = NUMBER OF BOXES NOT LOADED BY 'NEWLST'.
- LBOX = AN ARRAY OF (A-N) ELEMENTS EACH HAVING THE VALUE OF A BOX LENGTH OF THOSE BOXES LAST LOADED BY 'NEWLST'.
- SMALL = 4
- WIDE = LARGEST PALLET WIDTH REMAINING AFTER ALL THE BOXES WERE LOADED BY 'NEWLST'. THIS DOES NOT INCLUDE THE LAST ROW OF BOXES, IF MORE THAN ONE ROW IS LOADED BY

'NEWLST'.
MBL = ARRAY OF 'A' BOX LENGTHS, ALL POSITIVE IN VALUE.
MBW = ARRAY OF 'A' BOX WIDTHS, ALL POSITIVE IN VALUE.
COOR = AN ARRAY OF 'A' ELEMENTS. THE ENTRIES IN THIS ARRAY CORRESPONDING TO THE BOXES LOADED BY 'NEWLST' WILL HAVE A FIVE DIGIT COORDINATE OF THE UPPER LEFT CORNER OF EACH BOX. ALL OTHER ELEMENTS WILL REMAIN ZERO.
POINT = COORDINATE OF THE LAST BOX TO BE LOADED BY 'NEWLST'.
A = ORIGINAL NUMBER OF BOXES; A = NEXT+N.
STPT = 0

LINE(2): ESTABLISHES ALL VARIABLES AND ARRAYS TO BE INTEGER.

LINE(3): DIMENSION STATEMENT DECLARES ALL ARRAYS TO HAVE A MAXIMUM OF 30 ELEMENTS.

LINE(4): THE DATA STATEMENT INITIALIZES SEVERAL SINGLE VARIABLES AND ARRAYS TO BE ZERO.

LINE(5): 'TWIDTH' IS GIVEN THE VALUE OF 'WIDE'.

LINES(6) THRU (8): THIS DO LOOP DUPLICATES THE VALUES IN ARRAYS 'BOXLEN' AND 'BOXWTH' INTO ARRAYS 'MBL' AND 'MBW' RESPECTIVELY. SOME OF THESE BOXES IN 'BOXLEN' AND 'BOXWTH' WILL BE IDENTIFIED BY 'PILE4' AND 'PILE3' AS HAVING BEEN LOADED UNDER SOME BOX IN 'LBOX' (THOSE BOXES LAST LOADED BY 'NEWLST'). THEREFORE, LATER IN 'PILE4' SUBROUTINE 'REGRF' WILL BE CALLED TO MODIFY 'BOXLEN' AND 'BOXWTH'; ELIMINATING THESE LOADED BOXES.

LINE(9): STPT2 = STPT

LINES(10) THRU (13): THIS DO LOOP DETERMINES 'SWIDE'. 'SWIDE' IS THE REMAINING WIDTH OF THE PALLET WITH THE "LAST" ROW OF BOXES LOADED BY 'NEWLST' REMOVED.

LINE(14): THIS DO STATEMENT BEGINS THE MAJOR PORTION OF THE SUBROUTINE 'PILE4'. NOTE THE INDEX 'I' IS INCREMENTED BY ONE FROM ONE TO 'NEXT', THE NUMBER OF BOXES IN THE "LAST" ROW LOADED BY 'NEWLST'.

LINE(15): N1 IS INITIALIZED AS ZERO.

LINE(16): IF THE CURRENT VALUE OF 'I' IS LESS THAN 'START' THE PROGRAM SKIPS TO THE END OF THE DO LOOP AND THEN RETURNS TO LINE (14) WERE IT BEGINS WITH THE NEXT BOX. THE VALUE OF 'START' WILL BECOME OBVIOUS IN LINE (27).

LINE(18): 'TLENTH' IS GIVEN THE VALUE OF THE LENGTH OF THE I TH BOX IN THE "LAST" LOAD OF 'NEWLST'.

LINE(19): 'TWIDTH' IS ASSIGNED THE VALUE OF 'SWIDE' MINUS THE WIDTH OF THE I TH BOX IN TH "LAST" LOAD OF 'NEWLST'.

LINE(20): IF 'I' IS EQUAL TO 'NEXT' THE PROGRAM PROCEEDS TO LINE (31),

OTHERWISE IT CONTINUES TO THE NEXT STATEMENT.

LINE(22): IP=I + 1; THE CURRENT INDEX VALUE PLUS 1.

LINES(23) THRU (29): IN THIS DO LOOP ALL THE BOXES, BEGINNING WITH BOX(IP), ARE COMPARED TO THE WIDTH OF BOX(1). IF THERE IS LESS THAN FOUR INCHES DIFFERENCE 'TLENTH' IS INCREASED BY THE NEXT BOXES LENGTH, AND THE WIDTH IS DECREASED ACCORDINGLY. ALSO 'START' IS GIVEN THE VALUE OF THE INDEX CORRESPONDING TO THIS ADDITIONAL BOX. HOWEVER, WHEN THE FIRST BOX THAT IS ENCOUNTERED WITH A WIDTH GREATER THAN THE CURRENT BOX WIDTH (STORED IN TT2), THE PROGRAM EXITS THE DO LOOP AND PROCEEDS TO LINE (31). NOTE, WHAT THIS DO LOOP IS DOING IS TESTING FOR A "UNIFORM LOAD" ASSUMPTION FOR A "PORTION" OF THE "LAST" LOAD OF BOXES IDENTIFIED BY 'NEWLST', (SEE PARA. 5, "UNIFORM LOAD" ASSUMPTION).

LINE(30): IF THE DO LOOP IS COMPLETED, 'TWIDTH' IS SET EQUAL TO 'SWIDE' MINUS THE MAXIMUM BOX WIDTH OF THOSE BOXES IDENTIFIED IN LINES (23) THRU (29) AS HAVING WIDTH DIFFERENCES OF LESS THAN 4 INCHES.

LINE(31): 'SAVE' IS INCREMENTED BY 'TLENTH'.

LINE(32): IF 'I' IS EQUAL TO 'NEXT', THEN 'TLENTH' IS GIVEN THE VALUE OF 'LENGTH'.

LINES(34) THRU (38): THIS DO LOOP INITIALIZES SEVERAL ARRAYS TO ZERO.

LINES(39) THRU (41): THESE STATEMENTS INITIALIZES SEVERAL SINGLE VARIABLES TO ZERO.

LINE(42): 'NEWLST' IS CALLED. HOWEVER, THIS TIME THE LENGTH OF THE PALLET IS REPLACED BY 'TLENTH' AND THE WIDTH BY 'TWIDTH'. 'BOXLEN' AND 'BOXWTH' CONTAIN THOSE BOXES TO BE LOADED. 'MBL' AND 'MBW' STILL CONTAIN ALL THE ORIGINAL BOX LENGTHS AND WIDTHS. AS WELL, 'COOR' CONTAINS THE CURRENT COORDINATES OF ALL LOADED BOXES. ALL OTHER ARGUMENTS ARE ZERO OR HAVE THE VALUES ESTABLISHED WHEN 'PILE4' WAS CALLED.

LINE(43): WHEN 'NEWLST' RETURNS, IF 'N' IS ZERO, THAT IS, NO MORE BOXES ARE LEFT FOR LOADING THE PROGRAM RETURNS TO 'MAST3A'. IF 'N' IS NOT ZERO THE NEXT STATEMENT IS EXECUTED.

LINE(45): IF 'COUNT' IS EQUAL TO ZERO, THAT IS THERE WAS ONLY ONE BOX LOADED BY 'NEWLST', THE PROGRAM PROCEEDS TO LINE (49). IF 'COUNT' IS NOT ZERO, LINE (47) IS EXECUTED.

LINE(47): SUBROUTINE 'PILE3' IS CALLED. FOR AN INTERPRETATION OF EACH ARGUMENT SEE SECTION [I]. 'PILE3' CARRIES OUT THE STACKING PORTION OF THE PROGRAM UNDER EACH BOX THAT 'NEWLST' HAS JUST LOADED, (SEE PARA. 5, STEPS (6)(B), (7) AND (8)).

LINE (48): THE PROGRAM SKIPS TO LINE (54).

LINES(49) AND (50): IN LINE (45) IF ONLY ONE BOX IS LOADED BY 'NEWLST' THESE TWO STATEMENTS ARE EXECUTED NEXT. 'BOXL(1)' AND 'BOXW(1)' ARE

GIVEN THE LENGTH AND WIDTH OF THE FIRST BOX OR SERIES OF BOXES UNDER WHICH THE REMAINING BOXES ARE TO BE "STACKED" USING 'PILE3'.

LINE(51): SUBTOURTINE 'PILE3' IS CALLED. ALL ARGUMENTS HAVE THE SAME MEANING EXCEPT 'COUNT' IS REPLACED BY 1. 'PILE3' ALSO ASSIGNS THE APPROPRIATE COORDINATE TO THE BOXES THAT IT WILL LOAD.

LINE(52): IF 'I' IS EQUAL TO 'NEXT', THAT IS THE FINAL BOX IN THE "LAST" ROW OF BOXES ORIGINALLY IDENTIFIED BY 'NEWLST' HAS BEEN REACHED THE PROGRAM RETURNS TO 'MAST3A'.

LINE(54): SUBROUTINE 'REORP' IS CALLED. THIS SUBROUTINE WILL ELIMINATE THOSE BOXES LOADED BY 'NEWLST' AND 'PILE3' FROM THE LIST OF 'N' BOXES ORIGINALLY SENT TO 'PILE4'.

LINES(55) THRU (57): THESE STATEMENTS READJUST THE VALUE OF 'STPT2' TO CORRESPOND TO THE COORDINATE OF A POINT UNDER THE LOWER LEFT CORNER OF THE NEXT BOX IN ARRAY 'LBOX' (THE ARRAY OF BOXES SENT TO 'PILE4' AS

LINE(59): THIS STATEMENT RETURNS THE PROGRAM TO LINE(14) WHERE THE NEXT BOX IN 'LBOX' IS CHOSEN AND THE PROCEDURE IS PERFORMED AGAIN.

LINE(60): RETURN TO CALLING MODULE, 'MAST3A'.

LINE(61): END OF SUBROUTINE.

[I] SUBROUTINE PILE3

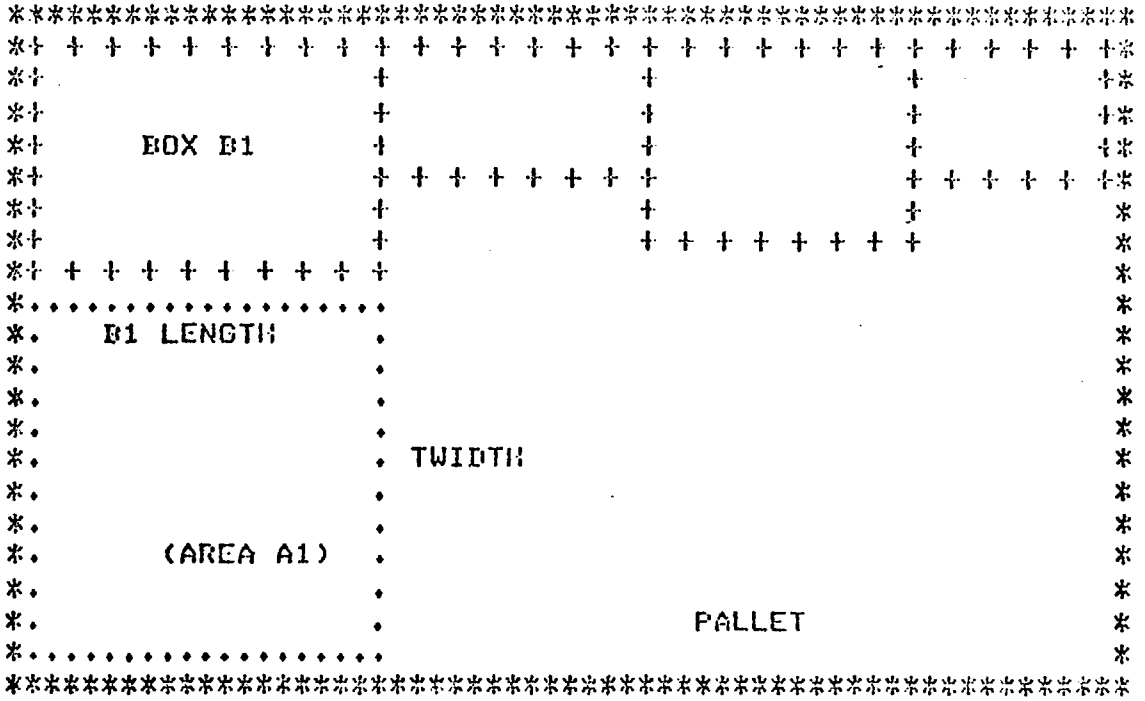
AS A REVIEW, REMEMBER WHEN 'MAST3A' CALLED 'NEWLST' A SERIES OF BOXES WERE LOADED ACROSS THE PALLETS "STARTING SIDE" UNTIL AN "UNEVEN" LOAD IS OBTAINED. NEXT, 'PILE4' IS CALLED. WITHIN 'PILE4' 'NEWLST' IS CALLED AGAIN. BUT THIS TIME, ONLY THE AREA (A1) UNDER BOX (B1) IN THE "LAST" ROW OF BOXES LOADED BY 'NEWLST' TAKES THE PLACE OF THE ENTIRE PALLET AREA (SEE FIG. 15). THE LENGTH OF (B1) IS USED AS THE LENGTH OF THIS AREA (A1) AND 'TWIDTH' IS THE WIDTH. OF COURSE THE "UNIFORM LOAD" ASSUMPTION IS ALSO TAKEN INTO EFFECT, SO THE LENGTH MAY BE THE SUM OF UP TO 'NEXT' NUMBER OF BOXES, WITH LENGTH EQUAL TO 'TLENTH' (SEE FIG. 15). ONCE 'NEWLST' HAS LOADED AS MANY BOXES IN AREA (A1) AS POSSIBLE, 'PILE3' IS CALLED TO PERFORM THE "STACKING" PORTION OF THE LOADING ROUTINE UNDER THE LAST SERIES OF BOXES LOADED BY 'NEWLST'.

IT IS OBVIOUS BY EXAMINING 'PILE3' AND 'PILE2A' THAT THE TWO SUBROUTINES ARE ALMOST IDENTICAL. HOWEVER, SOME SUBTLE CHANGES TO 'PILE3' HAVE BEEN MADE TO CONFORM TO ITS PARTICULAR REQUIREMENTS.

THEREFORE, INSTEAD OF DISCUSSING 'PILE3' IN ITS ENTIRETY, ONLY THOSE LINES THAT HAVE BEEN CHANGED OR ADDED OR DELETED IN 'PILE3' ARE DISCUSSED.

LINE(1): THE ARGUMENTS OF 'PILE3' HAVE THE SAME MEANING WITH THE FOLLOWING EXCEPTIONS;

LENGTH = 'TLENTH', CALCULATED IN 'PILE4'



AREA USED BY 'PILE3'

FIGURE 15

N3 = THE 'TOTAL' NUMBER OF BOXES NOT LOADED BY 'NEWLST' WHEN CALLED IN 'MAST3A' AND 'PILE4'.

BOXL = ARRAY OF (A-N) ELEMENTS EACH HAVING THE VALUE OF A BOX LENGTH OF THE 'TOTAL' NUMBER OF BOXES LOADED BY 'NEWLST'.

BOXW = ARRAY OF (A-N) ELEMENTS EACH HAVING THE VALUE OF A BOX WIDTH OF THE 'TOTAL' NUMBER OF BOXES LOADED BY 'NEWLST'.

TRACK = NUMBER OF BOXES LOADED BY 'NEWLST' IN 'PILE4'. THAT IS THE BOXES LOADED UNDER 'LENGTH'.

WIDE = LARGEST PALLET WIDTH REMAINING AFTER BOXES ARE LOADED BY 'NEWLST' WITHIN 'PILE4'.

POINT = COORDINATE OF THE LAST BOX TO BE LOADED BY 'NEWLST' IN 'PILE4'.

WITH THESE VARIABLE ADJUSTMENTS IN MIND ONLY SOME MINOR LINE CHANGES HAVE BEEN MADE TO 'PILE3'.

LINE(6): HAS BEEN CHANGED TO "WIDTH = WIDE" FROM "BPT = POINT" IN 'PILE2A', WHICH NO LONGER REQUIRED.

LINES (14), (47), (60), (64) AND (69) THAT WERE IN 'PILE2A', ARE ELIMINATED FROM 'PILE3'.

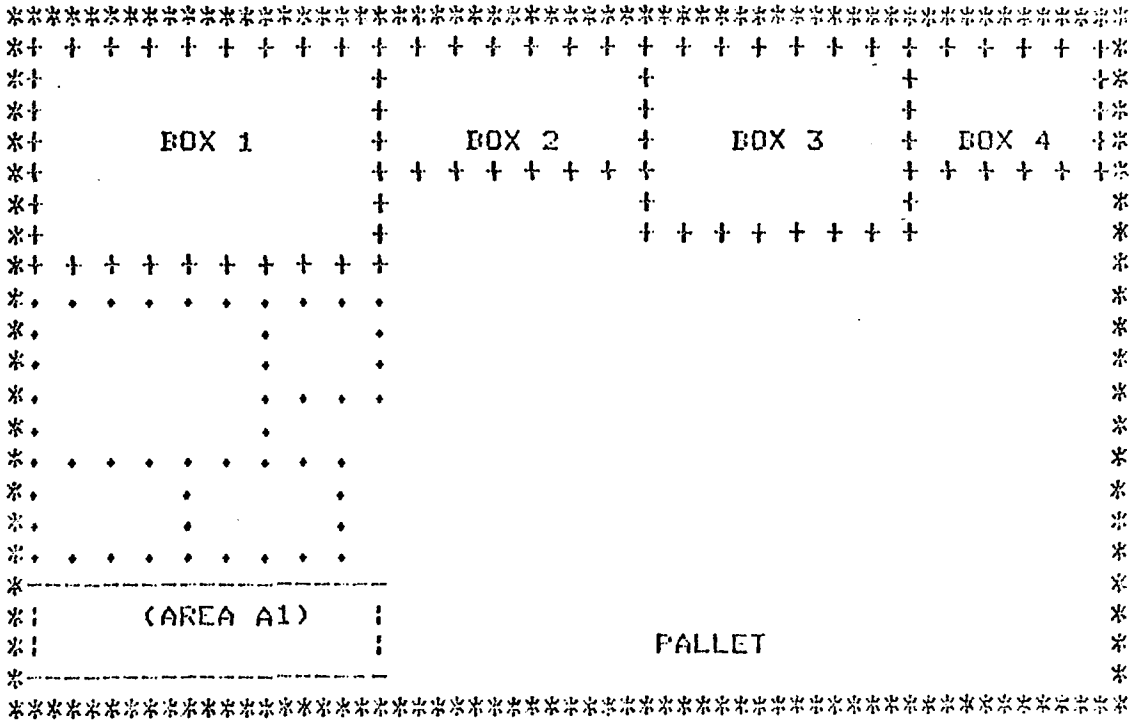
IN ADDITION, LINE (47) WAS ADDED TO 'PILE3'. CONSEQUENTLY, IN LINE (51) 'BOXLEN(K)' IS MADE A NEGATIVE VALUE, AND IN LINE (57) 'POINT' IS NOW INCREMENTED BY ADDING "100 X -BOXLEN(K)" INSTEAD OF JUST "BOXLEN(K)" AS IN 'PILE2A'.

[J] SUBROUTINE REORP

SUBROUTINE 'REORP' IS DESIGNED TO BE USED ONLY WITH CLASS 2 PROCEDURES, SUCH AS MODULE 'MAST3A'. SINCE 'REORP' IS WRITTEN IN A VERY CRYPTIC FORM AND BASICALLY PERFORMS A VARIABLE MANIPULATION, A LINE-BY-LINE DESCRIPTION WOULD BE VERY DIFFICULT. INSTEAD, A BRIEF OUTLINE FOLLOWS OF THE PURPOSE AND RESULTS OBTAINED FROM 'REORP'.

'REORP' STARTS WITH TWO ARRAYS, 'MBOXL' AND 'MBOXW', OF LENGTH 'N'. THESE TWO ARRAYS ARE PROVIDED BY 'PILE4' AND CONTAINS LENGTHS AND WIDTHS OF THOSE BOXES THAT WERE AVAILABLE FOR LOADING WHEN 'PILE4' WAS CALLED. ALSO INCLUDED IN THE CALLING ARGUMENTS OF 'REORP' ARE ARRAYS 'BOXLEN' AND 'BOXWTH' OF LENGTH 'N2'. THESE ARRAYS CONTAIN ONLY THE BOXES REMAINING FROM 'MBOXL' AND 'MBOXW' THAT WERE MADE "UNAVAILABLE" FOR FUTURE LOADS BY 'PILE4'. NOTE THAT MANY OF THE BOXES ELIMINATED IN 'MBOXL' AND 'MBOXW' WERE NOT LOADED BY 'PILE4'. INSTEAD, THESE ADDITIONAL BOXES WERE ELIMINATED BECAUSE THEY COULD NOT FIT IN THE REMAINING AREA (A1) ON THE PALLET (SEE FIG. 16). THUS, THESE BOXES MAY STILL BE CONSIDERED WHEN 'PILE4' BEGINS LOADING UNDER THE BOXES REMAINING IN THE "LAST" ROW OF BOXES LOADED BY 'NEWLST' WITHIN 'MAST3A'.

THE LAST TWO ARRAYS, EACH OF LENGTH 'N1' CONTAIN THOSE BOXES



REMAINING AREA AFTER ENTIRE SECOND LOAD

FIGURE 16

LOADED BY 'NEWLST' WHEN IT IS CALLED IN 'PILE4'.

SUBROUTINE 'REORP' IDENTIFIES THOSE BOXES THAT ARE STILL ELIGIBLE TO BE LOADED AND FINALLY REINSERTS THE LENGTHS IN ARRAYS 'BOXLEN' AND 'MBOXL', AND THE WIDTHS IN 'BOXWTH' AND 'MBOXW'. THESE ARRAYS ARE RETURNED TO 'PILE4', WHERE THEY ARE USED TO CONTINUE LOADING THE PALLET.

[K] SUBROUTINE SORT

THE SUBROUTINE 'SORT' IS USED IN EACH OF THE MODULE SUBROUTINES (I.E., MAST2A). 'SORT', LIKE 'REORP', IS QUITE SIMPLE IN TERMS OF THE COMPUTATIONS THAT ARE PERFORMED, BUT WRITTEN IN SUCH A CRYPTIC MANNER THAT A LINE-BY-LINE DISCUSSION WOULD NOT BE WORTHWHILE. THUS, ONLY A OUTLINE OF WHAT 'SORT' ACCOMPLISHES WILL BE PRESENTED.

IN EACH MODULE THE INITIAL SET OF BOX LENGTHS AND BOX WIDTHS ARE READ INTO ARRAYS 'BL' AND 'BW'. THESE ARRAYS MAY OR MAY NOT BE SORTED IN DESCENDING ORDER BY LENGTH. TO ACCOMPLISH THIS SORTING SUBROUTINE 'SORT' CREATES AN ARRAY, 'POINT' THAT CONTAINS AS MANY ELEMENTS AS 'BL' AND 'BW'. EACH ELEMENT OF 'POINT' CONTAINS THE ELEMENT NUMBER OR LOCATION OF THE NEXT LONGEST BOX. FOR EXAMPLE, IF 'POINT(15)' HAD A VALUE OF 22, THEN 'BL(22)' WILL BE THE NEXT LONGEST BOX.

TO BEGIN THIS PROCESS, 'SORT' MUST IDENTIFY A STARTING POINT. IN OTHER WORDS THE POSITION OF THE LONGEST BOX FROM THE ENTIRE SET OF 'N' BOXES. THIS VALUE IS RETURNED IN 'START'.

WHEN THE PROGRAM RETURNS TO WHICH EVER MODULE CALLED 'SORT' THE ARRAY 'POINT' AND THE VARIABLE 'START' CAN BE USED TO SORT ARRAYS 'BL' AND 'BW' IN DESCENDING ORDER ACCORDING TO BOX LENGTH.

BIBLIOGRAPHY

[1] DE SHA, C.L., "AREA-EFFICIENT AND VOLUME-EFFICIENT ALGORITHMS FOR LOADING CARGO," MASTERS THESIS, U.S. NAVAL POST GRADUATE SCHOOL, (1770).

[2] FISK, J.C. AND M.S. HUNG, "A HEURISTIC ROUTINE FOR SOLVING LARGE LOADING PROBLEMS," NAVAL RESEARCH QUARTELY, VOL. 26, PP. 643-649 (1979).

[3] HUNG, M.S. AND J.R. BROWN, "AN ALGORITHM FOR A CLASS OF LOADING PROBLEMS," NAVAL RESEARCH LOGISTICS QUARTERLY, VOL. 25, PP. 289-297 (1978).

[4] STEDEL, H.J., "GENERATING PALLET LOADING PATTERNS: A SPECIAL CASE OF THE TWO-DIMENSIONAL CUTTING STOCK PROBLEM," MANAGEMENT SCIENCE, VOL. 25, PP. 997-1004 (1979).

APPENDIX

I. PLR (PALLET LOADING ROUTINE)
II. MAST2A
III. MAST2B
IV. MAST2C
V. MAST2D
VI. MAST3A
VII. NEWLST
VIII. PILE2A
IX. PILE2B
X. PILE4
XI. PILE3
XII. SORT
XIII. REGRP

```
C*************************************************************************
C*                                                                       *
C*                    PALLET LOADING ROUTINE                             *
C*                                                                       *
C* THIS ROUTINE WILL RUN ALL LOADING ROUTINES, USING 'N'                 *
C* NUMBER OF BOXES FROM 'BAMS4.DAT'.  FOR EACH LOAD THE BOX              *
C* LENGTHS, WIDTHS AND UPPER LEFT HAND COORDINATE WILL BE                *
C* PRINTED.  FINALLY, THE NUMBER OF THE LARGEST LOADED AREA              *
C* WILL BE PRINTED AND THE OPTION THAT GAVE THIS AREA.                   *
C*                                                                       *
C* OPT       SUBROUTINES USED                                            *
C*                                                                       *
C* [1]       MAST2A,LINFIT,NEWLST,PILE2A,SORT                            *
C* [2]       MAST2B,LINFIT,NEWLST,PILE2B,SORT                            *
C* [3]       MAST2C,LINFIT,NEWLST,PILE2A,SORT                            *
C* [4]       MAST2D,LINFIT,NEWLST,PILE2B,SORT                            *
C* [5]       MAST3A,LINFIT,NEWLST,REGRP,PILE3&4,SORT                     *
C*                                                                       *
C*************************************************************************
0001          IMPLICIT INTEGER*2(A-Z)
0002          CALL ASSIGN(1,'BAMS4.DAT')
0003          CALL ASSIGN(5,'TI:')
0004          DEFINE FILE 1 (200,12,U,NEXT4)
0005          NAREA = 0
0006          DO 1 J=1,5
0007          READ(1'1) (N,I=1,12)
0008     2    AREA = 0
0009          IF(J.EQ.1) CALL MAST2A(N,AREA)
0011          IF(J.EQ.2) CALL MAST2B(N,AREA)
0013          IF(J.EQ.3) CALL MAST2C(N,AREA)
0015          IF(J.EQ.4) CALL MAST2D(N,AREA)
0017          IF(J.EQ.5) CALL MAST3A(N,AREA)
0019          IF(AREA.LE.NAREA) GO TO 1
0021          NAREA = AREA
0022          OPT   = J
0023          KHOICE = 99
0024     1    CONTINUE
0025          WRITE(5,1000) NAREA,OPT
0026  1000    FORMAT(//,' LARGEST LOADED AREA = ',I5,5X,'OPTION = ',I2)
0027          STOP
0028          END
```

```
C*************************************************************
C*                                                           *
C* THIS SUBROUTINE TAKES 'N' BOXES AND LOADS A PALLET OF     *
C* DIMENSION LENGTH = 104 BY WIDTH = 84. SUBROUTINES NEWLST/ *
C* LINFIT ARE USED INITIALLY TO COMPUTE ROW(S) OF BOXES THAT *
C* WILL FIT ALONG PALLET LENGTH. PILE 2A WILL THEN CONTINUE  *
C* LOADING ANY REMAINING BOXES UNDER EACH BOX IN THE LAST    *
C* ROW OF BOXES LOADED BY NEWLST. LOCATION OF THE UPPER LEFT *
C* CORNER OF EACH LOADED BOX IS COMPUTED BASED ON THE        *
C* COORDINATE SYSTEM (X,Y), STARTING WITH THE UPPER LEFT     *
C* HAND CORNER OF THE PALLET AS (1,1). NOTE, BOXES NOT       *
C* LOADED WILL HAVE COORDINATE = 0. OUTPUT IS THE LIST OF    *
C* 'N' BOXES WITH LOADED BOX LENGTHS NEGATIVE,  ALONG WITH
C* COORDINATES OF LOADED BOXES.                              *
C*                                                           *
C*************************************************************
0001        SUBROUTINE MAST2A(N,AREA)
0002        IMPLICIT INTEGER*2 (A-Z)
0003        DIMENSION BOXLEN(30),BOXWTH(30),BL(30),BW(30),PTLGH(30),
           *          LGH(360),WTH(360),LBOX(30),WBOX(30),TLBOX(30),
           *          TWBOX(30),MBL(30),MBW(30),COOR(30)
0004        DATA COOR,TLBOX,TWBOX,LBOX,WBOX,STLGH,N1,TOTAL,
           *     COUNT,TRACK,WIDE,SPACE,STPT/158*0/
0005        POINT  = 101
0006        DO 1 U=1,N
0007   1    READ(1'U) Z,Z,Z,Z,Z,BL(U),BW(U)
0008        CALL SORT(PTLGH,BL,STLGH,30)
0009        M = STLGH
0010        DO 2 K = 1,N
0011        BOXLEN(K) = BL(M)
0012        BOXWTH(K) = BW(M)
0013        MBL(K) = BL(M)
0014        MBW(K) = BW(M)
0015   2    M = PTLGH(M)
0016        A = N
0017        SMALL = 4
0018        CALL NEWLST(N,104,84,SPACE,BOXLEN,BOXWTH,SMALL,LBOX,
           *            WBOX,TRACK,WIDE,N1,TLBOX,TWBOX,
           *            MBL,MBW,COOR,POINT,A,STPT)
0019        IF(N.EQ.0) GO TO 10
0021        CALL PILE2A(BOXLEN,BOXWTH,104,N,LBOX,WBOX,TRACK,
           *           SMALL,WIDE,MBL,MBW,COOR,POINT,A)
0022   10   WRITE(5,1000)
0023   1000 FORMAT(//,1X,' LENGTH',3X,'WIDTH',3X,'POSITION',/)
0024        DO 3 S=1,A
0025        WRITE(5,2000) MBL(S),MBW(S),COOR(S)
0026   2000 FORMAT(1X,I4,5X,I4,5X,I5)
0027   3    CONTINUE
0028        DO 4 T=1,A
0029        IF(MBL(T).GT.0) GO TO 4
0031        TOTAL = TOTAL+1
0032        AREA  = AREA+IABS(MBL(T)*MBW(T))
0033   4    CONTINUE
0034        WRITE(5,3000) AREA,TOTAL
```

```
      . . .
0035   3000 FORMAT(1X,' AREA = ',I5,5X,'TOTAL NO. OF BOXES = ',I3)
0036        RETURN
0037        END
```

```
      C*************************************************************************
      C*                                                                       *
      C* THIS SUBROUTINE TAKES 'N' BOXES AND LOADS A PALLET OF                 *
      C* DIMENSION LENGTH = 104 BY WIDTH = 84. SUBROUTINES NEWLST/             *
      C* LINFIT ARE USED INITIALLY TO COMPUTE ROW(S) OF BOXES THAT             *
      C* WILL FIT ALONG PALLET LENGTH. PILE 2B WILL THEN CONTINUE              *
      C* LOADING ANY REMAINING BOXES UNDER EACH BOX IN THE LAST                *
      C* ROW OF BOXES LOADED BY NEWLST. LOCATION OF THE UPPER LEFT             *
      C* CORNER OF EACH LOADED BOX IS COMPUTED BASED ON THE                    *
      C* COORDINATE SYSTEM (X,Y), STARTING WITH THE UPPER LEFT                 *
      C* HAND CORNER OF THE PALLET AS (1,1). NOTE, BOXES NOT                   *
      C* LOADED WILL HAVE COORDINATE = 0. OUTPUT IS THE LIST OF                *
      C* 'N' BOXES WITH LOADED BOX LENGTHS NEGATIVE, ALONG WITH                *
      C* COORDINATES OF LOADED BOXES.                                          *
      C*                                                                       *
      C*************************************************************************
0001        SUBROUTINE MAST2B(N,AREA)
0002        IMPLICIT INTEGER*2 (A-Z)
0003        DIMENSION BOXLEN(30),BOXWTH(30),BL(30),BW(30),PTLGH(30),
           *          LGH(360),WTH(360),LBOX(30),WBOX(30),TLBOX(30),
           *          TWBOX(30),MBL(30),MBW(30),COOR(30)
0004        DATA COOR,TLBOX,TWBOX,LBOX,WBOX,STLGH,N1,TOTAL,COUNT,
           *     TRACK,WIDE,SPACE,STPT/158*0/
0005        POINT  = 101
0006        DO 1 U=1,N
0007    1   READ(1'U) Z,Z,Z,Z,Z,BL(U),BW(U)
0008        CALL SORT(PTLGH,BL,STLGH,30)
0009        M = STLGH
0010        DO 2 K = 1,N
0011        BOXLEN(K) = BL(M)
0012        BOXWTH(K) = BW(M)
0013        MBL(K) = BL(M)
0014        MBW(K) = BW(M)
0015    2   M = PTLGH(M)
0016        SMALL = 4
0017        A = N
0018        CALL NEWLST(N,104,84,SPACE,BOXLEN,BOXWTH,SMALL,LBOX,
           *            WBOX,TRACK,WIDE,N1,TLBOX,TWBOX,
           *            MBL,MBW,COOR,POINT,A,STPT)
0019        IF(N.EQ.0) GO TO 10
0021        CALL PILE2B(BOXLEN,BOXWTH,104,N,LBOX,WBOX,TRACK,
           *          SMALL,WIDE,MBL,MBW,COOR,POINT,A)
0022    10  WRITE(5,1000)
0023    1000 FORMAT(//,1X,' LENGTH',3X,'WIDTH',3X,'POSITION',/)
0024        DO 3 S=1,A
0025        WRITE(5,2000) MBL(S),MBW(S),COOR(S)
0026    2000 FORMAT(1X,I4,5X,I4,5X,I5)
0027    3   CONTINUE
0028        DO 4 T=1,A
0029        IF(MBL(T).GT.0) GO TO 4
0031        TOTAL = TOTAL+1
0032        AREA  = AREA+IABS(MBL(T)*MBW(T))
0033    4   CONTINUE
0034        WRITE(5,3000) AREA,TOTAL
```

```
0035    3000 FORMAT(1X,' AREA = ',I5,5X,'TOTAL NO. OF BOXES = ',I3)
0036         RETURN
0037         END
```

```
      C*************************************************************************
      C*                                                                       *
      C* THIS SUBROUTINE TAKES 'N' BOXES AND LOADS A PALLET OF                 *
      C* DIMENSION LENGTH = 84 BY WIDTH = 104. SUBROUTINES NEWLST/             *
      C* LINFIT ARE USED INITIALLY TO COMPUTE ROW(S) OF BOXES THAT             *
      C* WILL FIT ALONG PALLET LENGTH. PILE 2A WILL THEN CONTINUE              *
      C* LOADING ANY REMAINING BOXES UNDER EACH BOX IN THE LAST                *
      C* ROW OF BOXES LOADED BY NEWLST. LOCATION OF THE UPPER LEFT             *
      C* CORNER OF EACH LOADED BOX IS COMPUTED BASED ON THE                    *
      C* COORDINATE SYSTEM (X,Y), STARTING WITH THE UPPER LEFT                 *
      C* HAND CORNER OF THE PALLET AS (1,1). NOTE, BOXES NOT                   *
      C* LOADED WILL HAVE COORDINATE = 0. OUTPUT IS THE LIST OF                *
      C* 'N' BOXES WITH LOADED BOX LENGTHS NEGATIVE, ALONG WITH                *
      C* COORDINATES OF LOADED BOXES.                                          *
      C*                                                                       *
      C*************************************************************************
0001        SUBROUTINE MAST2C(N,AREA)
0002        IMPLICIT INTEGER*2 (A-Z)
0003        DIMENSION BOXLEN(30),BOXWTH(30),BL(30),BW(30),PTLGH(30),
           *          LGH(360),WTH(360),LBOX(30),WBOX(30),TLBOX(30),
           *          TWBOX(30),MBL(30),MBW(30),COOR(30)
0004        DATA COOR,TLBOX,TWBOX,LBOX,WBOX,STLGH,N1,TOTAL,COUNT,
           *     TRACK,WIDE,SPACE,STPT/158*0/
0005        POINT  = 101
0006        DO 1 U=1,N
0007   1    READ(1'U) Z,Z,Z,Z,Z,BL(U),BW(U)
0008        CALL SORT(PTLGH,BL,STLGH,30)
0009        M = STLGH
0010        DO 2 K = 1,N
0011        BOXLEN(K) = BL(M)
0012        BOXWTH(K) = BW(M)
0013        MBL(K) = BL(M)
0014        MBW(K) = BW(M)
0015   2    M = PTLGH(M)
0016        SMALL = 4
0017        A = N
0018        CALL NEWLST(N,84,104,SPACE,BOXLEN,BOXWTH,SMALL,LBOX,
           *            WBOX,TRACK,WIDE,N1,TLBOX,TWBOX,
           *            MBL,MBW,COOR,POINT,A,STPT)
0019        IF(N.EQ.0) GO TO 10
0021        CALL PILE2A(BOXLEN,BOXWTH,84,N,LBOX,WBOX,TRACK,
           *           SMALL,WIDE,MBL,MBW,COOR,POINT,A)
0022   10   WRITE(5,1000)
0023   1000 FORMAT(//,1X,' LENGTH',3X,'WIDTH',3X,'POSITION',/)
0024        DO 3 S=1,A
0025        WRITE(5,2000) MBL(S),MBW(S),COOR(S)
0026   2000 FORMAT(1X,I4,5X,I4,5X,I5)
0027   3    CONTINUE
0028        DO 4 T=1,A
0029        IF(MBL(T).GT.0) GO TO 4
0031        TOTAL = TOTAL+1
0032        AREA  = AREA+IABS(MBL(T)*MBW(T))
0033   4    CONTINUE
0034        WRITE(5,3000) AREA,TOTAL
```

```
0035    3000 FORMAT(1X,' AREA = ',I5,5X,'TOTAL NO. OF BOXES = ',I3)
0036         RETURN
0037         END
```

```
C*************************************************************
C*                                                           *
C* THIS SUBROUTINE TAKES 'N' BOXES AND LOADS A PALLET OF     *
C* DIMENSION LENGTH = 84 BY WIDTH = 104. SUBROUTINE NEWLST/  *
C* LINFIT ARE USED INITIALLY TO COMPUTE ROW(S) OF BOXES THAT *
C* WILL FIT ALONG PALLET LENGTH. PILE 2B WILL THEN CONTINUE  *
C* LOADING ANY REMAINING BOXES UNDER EACH BOX IN THE LAST    *
C* ROW OF BOXES LOADED BY NEWLST. LOCATION OF THE UPPER LEFT *
C* CORNER OF EACH LOADED BOX IS COMPUTED BASED ON THE        *
C* COORDINATE SYSTEM (X,Y), STARTING WITH THE UPPER LEFT     *
C* HAND CORNER OF THE PALLET AS (1,1). NOTE, BOXES NOT       *
C* LOADED WILL HAVE COORDINATE = 0. OUTPUT IS THE LIST OF    *
C* 'N' BOXES WITH LOADED BOX LENGTHS NEGATIVE, ALONG WITH    *
C* COORDINATES OF LOADED BOXES.                              *
C*                                                           *
C*************************************************************
      SUBROUTINE MAST2D(N,AREA)
      IMPLICIT INTEGER*2 (A-Z)
      DIMENSION BOXLEN(30),BOXWTH(30),BL(30),BW(30),PTLGH(30),
     *          LGH(360),WTH(360),LBOX(30),WBOX(30),TLBOX(30),
     *          TWBOX(30),MBL(30),MBW(30),COOR(30)
      DATA COOR,TLBOX,TWBOX,LBOX,WBOX,STLGH,N1,TOTAL,COUNT,
     *     TRACK,WIDE,SPACE,STPT/158*0/
      POINT  = 101
      DO 1 U=1,N
    1 READ(1'U) Z,Z,Z,Z,Z,BL(U),BW(U)
      CALL SORT(PTLGH,BL,STLGH,30)
      M = STLGH
      DO 2 K = 1,N
      BOXLEN(K) = BL(M)
      BOXWTH(K) = BW(M)
      MBL(K) = BL(M)
      MBW(K) = BW(M)
    2 M = PTLGH(M)
      SMALL = 4
      A = N
      CALL NEWLST(N,84,104,SPACE,BOXLEN,BOXWTH,SMALL,LBOX,
     *            WBOX,TRACK,WIDE,N1,TLBOX,TWBOX,
     *            MBL,MBW,COOR,POINT,A,STPT)
      IF(N.EQ.0) GO TO 10
      CALL PILE2B(BOXLEN,BOXWTH,84,N,LBOX,WBOX,TRACK,
     *            SMALL,WIDE,MBL,MBW,COOR,POINT,A)
   10 WRITE(5,1000)
 1000 FORMAT(//,1X,' LENGTH',3X,'WIDTH',3X,'POSITION',/)
      DO 3 S=1,A
      WRITE(5,2000) MBL(S),MBW(S),COOR(S)
 2000 FORMAT(1X,I4,5X,I4,5X,I5)
    3 CONTINUE
      DO 4 T=1,A
      IF(MBL(T).GT.0) GO TO 4
      TOTAL = TOTAL+1
      AREA  = AREA+IABS(MBL(T)*MBW(T))
    4 CONTINUE
      WRITE(5,3000) AREA,TOTAL
```

```
0035    3000 FORMAT(1X,' AREA = ',I5,5X,'TOTAL NO. OF BOXES = ',I3)
0036         RETURN
0037         END
```

```
C*********************************************************************
C*                                                                   *
C* THIS SUBROUTINE TAKES 'N' BOXES AND LOADS A PALLET OF             *
C* DIMENSION LENGTH = 104 BY WIDTH = 84. SUBROUTINES NEWLST/         *
C* LINFIT ARE USED INITIALLY TO COMPUTE ROW(S) OF BOXES THAT         *
C* WILL FIT ALONG PALLET LENGTH.  PILE4 WILL THE CONTINUE            *
C* LOADING ANY REMAINING BOXES UNDER EACH BOX IN THE LAST            *
C* ROW OF BOXES LOADED BY NEWLST, BY USING SUBROUTINES PILE3          *
C* AND NEWLST. LOCATION OF THE UPPER LEFT CORNER OF EACH             *
C* LOADED BOX IS COMPUTED BASED ON THE COORDINATE SYSTEM             *
C* (X,Y), STARTING WITH THE UPPER LEFT HAND CORNER OF THE            *
C* PALLET AS (1,1). NOTE, BOXES NOT LOADED WILL HAVE A               *
C* COORDINATE = 0. OUTPUT IS THE LIST OF 'N' BOXES WITH              *
C* LOADED BOX LENGTH NEGATIVE, ALONG WITH COORDINATES OF             *
C* THE LOADED BOXES.                                                 *
C*                                                                   *
C*********************************************************************
0001        SUBROUTINE MAIN1A(N,AREA)
0002        IMPLICIT INTEGER*2 (A-Z)
0003        DIMENSION BOXLEN(30),BOXWTH(30),BL(30),BW(30),PTLGH(30),
           *          LGH(30),WTH(30),LBOX(30),WBOX(30),TLBOX(30),
           *          TWBOX(30),MBL(30),MBW(30),COOR(30)
0004        DATA COOR,TLBOX,TWBOX,LBOX,WBOX,STLGH,N1,TOTAL,COUNT,
           *     TRACK,WIDE,SPACE,STPT/158*0/
0005        POINT  = 101
0006        DO 1 U=1,N
0007   1    READ(1'U) Z,Z,Z,Z,BL(U),BW(U)
0008        CALL SORT(PTLGH,BL,STLGH,30)
0009        M = STLGH
0010        DO 2 K = 1,N
0011        BOXLEN(K) = BL(M)
0012        BOXWTH(K) = BW(M)
0013        MBL(K) = BL(M)
0014        MBW(K) = BW(M)
0015   2    M = PTLGH(M)
0016        SMALL = 4
0017        A = N
0018        CALL NEWLST(N,104,84,SPACE,BOXLEN,BOXWTH,SMALL,LBOX,
           *            WBOX,TRACK,WIDE,N1,TLBOX,TWBOX,
           *            MBL,MBW,COOR,POINT,A,STPT)
0019        IF(N.EQ.0) GO TO 10
0021        CALL PILE4(BOXLEN,BOXWTH,104,N,LBOX,WBOX,TRACK,
           *           SMALL,WIDE,MBL,MBW,COOR,POINT,A,STPT)
0022   10   WRITE(5,1000)
0023   1000 FORMAT(//,' LENGTH',3X,'WIDTH',3X,'POSITION',/)
0024        DO 3 S=1,A
0025        WRITE(5,2000) MBL(S),MBW(S),COOR(S)
0026   2000 FORMAT(1X,I4,5X,I4,5X,I5)
0027   3    CONTINUE
0028        DO 4 T=1,A
0029        IF(MBL(T).GT.0) GO TO 4
0031        TOTAL = TOTAL+1
0032        AREA = AREA+IABS(MBW(T)*MBL(T))
0033   4    CONTINUE
```

```
0034          WRITE(5,3000) AREA,TOTAL
0035     3000 FORMAT(1X,' AREA = ',I5,5X,'TOTAL NO. OF BOXES = ',I3)
0036          RETURN
0037          END
```

```
      C*************************************************************
      C*                                                           *
      C* THIS SUBROUTINE STARTS WITH 'TRACK' NUMBER OF BOXES       *
      C* WITH LENGTHS, LBOX() AND WIDTHS, WBOX() THAT WERE         *
      C* IDENTIFIED BY NEWLST.                                     *
      C* THEN, AS MANY ADDITIONAL BOXES FROM A LIST OF 'N'         *
      C* BOXES ARE LOADED BELOW EACH SUCCESSIVE LBOX().            *
      C*                                                           *
      C*************************************************************
0001        SUBROUTINE PILE2A (BOXLEN,BOXWTH,LENGTH,N,LBOX,WBOX,TRACK,
     *                         SMALL,WIDE,MBL,MBW,COOR,POINT,A)
0002        IMPLICIT INTEGER*2(A-Z)
0003        DIMENSION BOXLEN(30),BOXWTH(30),LBOX(30),WBOX(30),
     *             COOR(30),MBL(30),MBW(30)
0004        SAVE   = 0
0005        START  = 0
0006        SWIDE  = 0
0007        BPT    = POINT
      C*************************************************************
      C*     LOAD ADDITIONAL BOXES BELOW EACH BOX IN LAST ROW      *
      C*     OF LINFIT BOXES.                                      *
      C*************************************************************
0008        DO 1 L=1,TRACK
0009        IF((WBOX(L)+WIDE).GT.SWIDE) SWIDE = WBOX(L)+WIDE
0011      1 CONTINUE
0012        TT2 = 0
0013        DO 2 I = 1,TRACK
0014        POINT = BPT
0015        TEMP = 0
0016        IF(I.LE.START) GO TO 2
0018        NEWLEN = LBOX(I)
0019        NEWWTH = SWIDE-WBOX(I)
0020        IF(I.EQ.TRACK) GO TO 10
0022        IP1    = I+1
0023        DO 3 M = IP1,TRACK
0024        IF(IABS(WBOX(M)-WBOX(I)).GT.4) GO TO 10
0026        NEWLEN = NEWLEN+LBOX(M)
0027        TT  = MAXO(WBOX(M),WBOX(M-1))
0028        TT2 = MAXO(TT2,TT)
0029        START  = M
0030    3   CONTINUE
0031        NEWWTH = SWIDE-TT2
0032   10   SAVE   = SAVE+NEWLEN
0033        IF(I.EQ.TRACK) NEWLEN = NEWLEN+LENGTH-SAVE
0035   15   MAX    = 0
0036        WMAX   = 0
0037        TEMPT = 0
0038        DO 4 K = 1,N
0039        IF(BOXLEN(K).LT.0.OR.BOXLEN(K).GT.NEWLEN.OR.BOXWTH(K).GT.
     *    NEWWTH) GO TO 4
0041        NEWWTH = NEWWTH-BOXWTH(K)
0042        IF(NEWLEN-BOXLEN(K).LT.SMALL) GO TO 20
0044        IF(BOXLEN(K).LT.MAX) GO TO 25
0046        MAX    = BOXLEN(K)
```

```
0047          TEMPT = POINT+100*MAX
0048          GO TO 25
0049    20    WMAX  = WMAX+BOXWTH(K)
      C*************************************************
      C* ASSIGN THE UPPER LEFT CORNER OF ALL LOADED BOXES A   *
      C* COORDINATE (X,Y).                                    *
      C*************************************************
0050    25    DO 5 C = 1,A
0051          IF(MBL(C).LT.0) GO TO 5
0053          IF(MBL(C).NE.BOXLEN(K).OR.MBW(C).NE.BOXWTH(K)) GO TO 5
0055          MBL(C) = -MBL(C)
0056          COOR(C) = POINT
0057          GO TO 30
0058  5       CONTINUE
0059  30      POINT = POINT+BOXWTH(K)
0060          BOXLEN(K) = -BOXLEN(K)
0061    4     CONTINUE
0062          IF(MAX.EQ.0) GO TO 35
0064          POINT = TEMPT
0065          NEWLEN = NEWLEN-MAX
0066          NEWWTH = SWIDE-WBOX(I)-WMAX-TEMP
0067          TEMP   = WMAX
0068          GO TO 15
0069    35    IF(I.NE.TRACK) BPT = BPT+LBOX(I)*100-WBOX(I)+WBOX(I+1)
0071    2     CONTINUE
0072          RETURN
0073          END
```

```
      C*************************************************************
      C*                                                           *
      C* THIS SUBROUTINE STARTS WITH 'TRACK' NUMBER OF BOXES       *
      C* WITH LENGTHS, LBOX() AND WIDTHS, WBOX() THAT WERE         *
      C* IDENTIFIED BY SUBROUTINE NEWLST.                          *
      C* THEN, AS MANY ADDITIONAL BOXES FROM A LIST OF 'N'         *
      C* BOXES ARE LOADED BELOW EACH SUCCESSIVE LBOX().            *
      C*                                                           *
      C*************************************************************
0001        SUBROUTINE PILE2B (BOXLEN,BOXWTH,LENGTH,N,LBOX,WBOX,TRACK,
     *                         SMALL,WIDE,MBL,MBW,COOR,POINT,A)
0002        IMPLICIT INTEGER*2(A-Z)
0003        DIMENSION BOXLEN(30),BOXWTH(30),LBOX(30),WBOX(30),
     *             COOR(30),MBL(30),MBW(30)
0004        SAVE   = 0
0005        START  = 0
0006        SWIDE  = 0
0007        BPT    = POINT
      C*************************************************************
      C*       LOAD ADDITIONAL BOXES BELOW EACH BOX IN LAST ROW    *
      C*       OF LINFIT BOXES.                                    *
      C*************************************************************
0008        DO 1 L=1,TRACK
0009        IF((WBOX(L)+WIDE).GT.SWIDE) SWIDE = WBOX(L)+WIDE
0011    1   CONTINUE
0012        TT2 = 0
0013        DO 2 I = 1,TRACK
0014        POINT = BPT
0015        TEMP = 0
0016        IF(I.LE.START) GO TO 2
0018        NEWLEN = LBOX(I)
0019        NEWWTH = SWIDE-WBOX(I)
0020        IF(I.EQ.TRACK) GO TO 10
0022        IP1    = I+1
0023        DO 3 M = IP1,TRACK
0024        IF(IABS(WBOX(M)-WBOX(I)).GT.4) GO TO 10
0026        NEWLEN = NEWLEN+LBOX(M)
0027        TT  = MAX0(WBOX(M),WBOX(M-1))
0028        TT2 = MAX0(TT2,TT)
0029        START  = M
0030    3   CONTINUE
0031        NEWWTH = SWIDE-TT2
0032    10  SAVE   = SAVE+NEWLEN
0033        IF(I.EQ.TRACK) NEWLEN = NEWLEN+LENGTH-SAVE
0035    15  MAX    = 0
0036        WMAX   = 0
0037        TEMPT = 0
0038        DO 4 K = 1,N
0039        IF(BOXLEN(K).LT.0.OR.BOXWTH(K).GT.NEWLEN.OR.BOXLEN(K).GT.
     *    NEWWTH) GO TO 4
0041        NEWWTH = NEWWTH-BOXLEN(K)
0042        IF(NEWLEN-BOXWTH(K).LT.SMALL) GO TO 20
0044        IF(BOXWTH(K).LT.MAX) GO TO 25
0046        MAX    = BOXWTH(K)
```

```
0047          TEMPT = POINT+100*MAX
0048          GO TO 25
0049     20   WMAX  = WMAX+BOXLEN(K)
         C**************************************************
         C* ASSIGN THE UPPER LEFT CORNER OF ALL LOADED BOXES A  *
         C* COORDINATE (X,Y).                                  *
         C**************************************************
0050     25   DO 5 C = 1,A
0051          IF(MBL(C).LT.0) GO TO 5
0053          IF(MBL(C).NE.BOXLEN(K).OR.MBW(C).NE.BOXWTH(K)) GO TO 5
0055          MBL(C) = -MBL(C)
0056          COOR(C) = POINT
0057          GO TO 30
0058     5    CONTINUE
0059     30   POINT = POINT+BOXLEN(K)
0060          BOXLEN(K) = -BOXLEN(K)
0061     4    CONTINUE
0062          IF(MAX.EQ.0) GO TO 35
0064          POINT = TEMPT
0065          NEWLEN = NEWLEN-MAX
0066          NEWWTH = SWIDE-WBOX(I)-WMAX-TEMP
0067          TEMP   = WMAX
0068          GO TO 15
0069     35   IF(I.NE.TRACK) BPT = BPT+LBOX(I)*100-WBOX(I)+WBOX(I+1)
0071     2    CONTINUE
0072          RETURN
0073          END
```

```
      C*************************************************************
      C*                                                           *
      C* THIS SUBROUTINE STARTS WITH 'TRACK' NUMBER OF BOXES,      *
      C* WITH LENGTHS, BOXL() AND WIDTHS, BOXW() THAT WERE         *
      C* IDENTIFIED BY SUBROUTINE PILE4.                           *
      C* THEN, AS MANY ADDITIONAL BOXES FROM A LIST OF 'N3'        *
      C* BOXES ARE LOADED BELOW EACH SUCCESSIVE BOXL().            *
      C*                                                           *
      C*************************************************************
0001        SUBROUTINE PILE3 (BOXLEN,BOXWTH,LENGTH,N3,BOXL,BOXW,
     *                    TRACK,SMALL,WIDE,POINT,MBL,MBW,COOR,A)
0002        IMPLICIT INTEGER*2(A-Z)
0003        DIMENSION BOXLEN(30),BOXWTH(30),BOXL(30),BOXW(30),
     *             MBL(30),MBW(30),COOR(30)
0004        SAVE   = 0
0005        START  = 0
0006        SWIDE  = 0
0007        WIDTH  = WIDE
      C*************************************************************
      C*    LOAD ADDITIONAL BOXES BELOW EACH BOX IN LAST ROW       *
      C*    OF LINFIT BOXES                                        *
      C*************************************************************
0008        DO 1 T=1,TRACK
0009        IF((BOXW(T)+WIDE).GT.SWIDE) SWIDE = BOXW(T)+WIDE
0011   1    CONTINUE
0012        TT2 = 0
0013        DO 2 I = 1,TRACK
0014        TEMP   = 0
0015        IF(I.LE.START) GO TO 2
0017        NEWLEN = BOXL(I)
0018        NEWWTH = SWIDE-BOXW(I)
0019        IF(I.EQ.TRACK) GO TO 10
0021        IP1    = I+1
0022        DO 3 M = IP1,TRACK
0023        IF(IABS(BOXW(M)-BOXW(I)).GT.4) GO TO 10
0025        NEWLEN = NEWLEN+BOXL(M)
0026        TT = MAX0(BOXW(M),BOXW(M-1))
0027        TT2 = MAX0(TT2,TT)
0028        START  = M
0029   3    CONTINUE
0030        NEWWTH = SWIDE-TT2
0031   10   SAVE   = SAVE+NEWLEN
0032        IF(I.EQ.TRACK) NEWLEN = NEWLEN+LENGTH-SAVE
0034   15   MAX    = 0
0035        WMAX   = 0
0036        DO 4 K = 1,N3
0037        IF(BOXLEN(K).LT.0.OR.BOXWTH(K).GT.NEWLEN.OR.BOXLEN(K).GT.
     *   NEWWTH) GO TO 4
0039        NEWWTH = NEWWTH-BOXLEN(K)
0040        IF(NEWLEN-BOXWTH(K).LT.SMALL) GO TO 20
0042        IF(BOXWTH(K).LT.MAX) GO TO 25
0044        MAX    = BOXWTH(K)
0045        GO TO 25
0046   20   WMAX   = WMAX+BOXLEN(K)
```

```
0047    25    BOXLEN(K) = -BOXLEN(K)
      C*************************************************************
      C* ASSION THE UPPER LEFT CORNER OF ALL LOADED BOXES A       *
      C* COORDINATE (X,Y).                                        *
      C*************************************************************
0048        DO 5 R=1,A
0049        IF(MBL(R).LT.0) GO TO 5
0051        IF(MBL(R).NE.-BOXLEN(K).OR.MBW(R).NE.BOXWTH(K)) GO TO 5
0053        MBL(R) = -MBL(R)
0054        COOR(R) = POINT
0055        GO TO 30
0056    5   CONTINUE
0057    30  POINT = POINT+100*-BOXLEN(K)
0058    4   CONTINUE
0059        IF(MAX.EQ.0) GO TO 2
0061        NEWLEN = NEWLEN-MAX
0062        NEWWTH = WIDTH-BOXW(I)-WMAX-TEMP
0063        TEMP   = WMAX
0064        GO TO 15
0065    2   CONTINUE
0066        RETURN
0067        END
```

```
C*************************************************************
C*                                                           *
C* THIS SUBROUTINE STARTS WITH 'N' NUMBER OF BOXES           *
C* WITH LENGTHS, 'BOXLEN' AND WIDTHS, 'BOXWTH'.  IT WILL     *
C* THEN LOAD, FROM THIS GROUP OF BOXES, BELOW THOSE BOXES    *
C* THAT WERE IDENTIFIED BY NEWLST.                           *
C* THIS LOADING WILL TAKE PLACE WITH THE HELP OF SUBROUT-    *
C* INES NEWLST AND PILE3.                                    *
C*                                                           *
C*************************************************************
0001        SUBROUTINE PILE4(BOXLEN,BOXWTH,LENGTH,N,LBOX,WBOX,NEXT,
           *                 SMALL,WIDE,MBL,MBW,COOR,POINT,A,STPT)
0002        IMPLICIT INTEGER*2(A-Z)
0003        DIMENSION BOXLEN(30),BOXWTH(30),LBOX(30),WBOX(30),BOXL(30),
           *                 BOXW(30),TLBOX(30),MBOXL(30),MBOXW(30),
           *                 TWBOX(30),MBL(30),MBW(30),COOR(30)
0004        DATA SWIDE,SUBTOT,SAVE,START,N1,MM,NN,TT2/8*0/
0005        TWIDTH = WIDE
0006        DO 1 M=1,N
0007        MBOXL(M) = BOXLEN(M)
0008   1    MBOXW(M) = BOXWTH(M)
0009        STPT2  = STPT
0010        DO 2 M=1,NEXT
0011        IF((WBOX(M)+WIDE).GT.SWIDE) SWIDE=WBOX(M)+WIDE
0013   2    CONTINUE
0014        DO 3 I = 1,NEXT
0015        N1 = 0
0016        IF(I.LE.START) GO TO 3
0018        TLENTH = LBOX(I)
0019        TWIDTH = SWIDE-WBOX(I)
0020        IF(I.EQ.NEXT) GO TO 10
0022        IP1    = I+1
0023        DO 4 J = IP1,NEXT
0024        IF(IABS(WBOX(J)-WBOX(I)).GT.4) GO TO 10
0026        TLENTH = TLENTH+LBOX(J)
0027        TT  = MAXO(WBOX(J),WBOX(J-1))
0028        TT2 = MAXO(TT2,TT)
0029   4    START = J
0030        TWIDTH = SWIDE-TT2
0031  10    SAVE   = SAVE+TLENTH
0032        IF(I.EQ.NEXT) TLENTH = TLENTH+LENGTH-SAVE
0034        DO 5 K = 1,30
0035        TLBOX(K) = 0
0036        TWBOX(K) = 0
0037        BOXL(K)   = 0
0038  5     BOXW(K)   = 0
0039        SPACE  = 0
0040        TWIDE  = 0
0041        COUNT  = 0
0042        CALL NEWLST(N,TLENTH,TWIDTH,SPACE,BOXLEN,BOXWTH,SMALL,BOXL,
           *             BOXW,COUNT,TWIDE,N1,TLBOX,TWBOX,MBL,MBW,COOR,
           *             POINT,A,STPT)
0043        IF(N.EQ.0) RETURN
0045        IF(COUNT.EQ.0) GO TO 15
```

```
0047          CALL PILE3(BOXLEN,BOXWTH,TLENTH,N2,BOXL,BOXW,
             *             COUNT,SMALL,TWIDE,POINT,MBL,MBW,COOR,A)
0048          GO TO 20
0049    15    BOXL(1) = LBOX(I)
0050          BOXW(1) = WBOX(I)
0051          CALL PILE3(BOXLEN,BOXWTH,TLENTH,N2,BOXL,BOXW,
             *             1,SMALL,TWIDE,POINT,MBL,MBW,COOR,A)
0052          IF(I.EQ.NEXT) RETURN
0054    20    CALL REGRP(BOXLEN,BOXWTH,MBOXL,MBOXW,TLBOX,TWBOX,
             *             N,N1,N2)
0055          IF((I+1).LE.NEXT) POINT = STPT2+WBOX(I+1)
0057          IF((I+1).LE.NEXT.AND.(I+1).NE.2) STPT2 = STPT2+100*LBOX(I+1)
0059    3     CONTINUE
0060          RETURN
0061          END
```

```
      C*************************************************************
      C*                                                           *
      C* THIS ROUTINE TAKES A LIST OF NUMBERS "LIST" OF LENGTH     *
      C* "AMT" AND CREATES A POINTER WHICH ORDERS THE NUMBERS FROM *
      C* LARGEST TO SMALLEST. THIS POINTER IS RETURNED IN THE      *
      C* VECTOR "POINT" ALONG WITH THE STARTING POSITION "START".  *
      C*                                                           *
      C*************************************************************
0001        SUBROUTINE SORT(POINT,LIST,START,AMT)
0002        IMPLICIT INTEGER*2 (A-Z)
0003        DIMENSION POINT(AMT),LIST(AMT)
0004        LAST  = 0
0005        NEXT  = 0
0006        AGAIN = 0
0007   1    TEMP  = 0
0008        DO 2 I = 1,AMT
0009        IF(POINT(I).NE.0    ) GO TO 2
0011        IF(LIST(I) .LE.TEMP ) GO TO 2
0013        IF(LIST(I) .EQ.AGAIN) GO TO 2
0015        TEMP = LIST(I)
0016        NEXT = I
0017   2    CONTINUE
0018        IF(LAST.EQ.0) GO TO 3
0020        POINT(LAST) = NEXT
0021   3    LAST = NEXT
0022        IF(NEXT.NE.AMT) GO TO 4
0024        BEGIN = NEXT
0025        GO TO 5
0026   4    BEGIN = NEXT+1
0027   5    DO 6 K = BEGIN,AMT
0028        IF(LIST(K).NE.LIST(LAST)) GO TO 6
0030        POINT(LAST) = K
0031        LAST = K
0032   6    CONTINUE
0033        AGAIN = TEMP
0034        COUNT = 0
0035        DO 7 J = 1,AMT
0036        IF(POINT(J).NE.0) GO TO 7
0038        COUNT = COUNT+1
0039        IF(COUNT.GE.2) GO TO 1
0041   7    CONTINUE
0042        TOP = 0
0043        DO 8 M = 1,AMT
0044        IF (LIST(M).LE.TOP) GO TO 8
0046        START = M
0047        TOP = LIST(M)
0048   8    CONTINUE
0049        RETURN
0050        END
```

```
      C*************************************************************
      C*                                                           *
      C* NEWLST BEGINS WITH 'N' BOXES OF LENGTH BOXLEN() AND       *
      C* RUNS SUCCESSIVE LINFIT SUBROUTINES ALONG THE LINEAR       *
      C* DISTANCE 'LENGTH'.  IT RETURNS LOADED BOXES WITH          *
      C* THEIR BOXLEN VALUES NEGATIVE.                             *
      C*                                                           *
      C*************************************************************
0001        SUBROUTINE NEWLST(N,LENGTH,WIDTH,SPACE,BOXLEN,BOXWTH,
     *                    SMALL, LBOX,WBOX,TRACK,WIDE,N1,TLBOX,
     *                    TWBOX,MBL,MBW,COOR,POINT,A,STPT)
0002        IMPLICIT INTEGER*2 (A-Z)
0003        DIMENSION BOXLEN(30),BOXWTH(30),LBOX(30),WBOX(30),LLBOX(30),
     *            WWBOX(30),TLBOX(30),MBL(30),MBW(30),COOR(30),
     *            TWBOX(30)
0004        NEWWTH = WIDTH
0005     10 DO 1 K=1,N
0006        IF(BOXLEN(K).LT.0)GO TO 1
0008        IF(BOXWTH(K).LE.NEWWTH) GO TO 1
0010        BOXWTH(K) = -BOXWTH(K)
0011        BOXLEN(K) = -BOXLEN(K)
0012      1 CONTINUE
0013        SPACE = 0
0014        CALL LINFIT(N,LENGTH,SPACE,BOXLEN)
0015        COUNT = 0
0016        TRACK = 0
0017        NUM = 0
0018        DO 2 L=1,N
0019        IF(BOXLEN(L).LT.0.AND.BOXWTH(L).LT.0) GO TO 2
0021        IF(BOXLEN(L).GT.0) GO TO 15
0023        TRACK = TRACK+1
0024        LBOX(TRACK) = -BOXLEN(L)
0025        WBOX(TRACK) = IABS(BOXWTH(L))
0026        N1 = N1+1
0027        TLBOX(N1) = -BOXLEN(L)
0028        TWBOX(N1) = IABS(BOXWTH(L))
0029        GO TO 2
0030     15 IF(BOXWTH(L).LT.0) GO TO 2
0032        NUM = NUM+1
0033        LLBOX(NUM) = BOXLEN(L)
0034        WWBOX(NUM) = BOXWTH(L)
0035      2 CONTINUE
0036        N = NUM
0037        IF(TRACK.EQ.0) RETURN
0039        IF(NUM.EQ.0) RETURN
0041        NEWLEN = LBOX(1)
0042        WIDE   = NEWWTH
0043        IF(TRACK.EQ.1) GO TO 25
0045        DIFF = 0
0046        SDIFF = 0
0047        DO 3 M=2,TRACK
0048        IF((WBOX(M)-WBOX(1)).LT.SDIFF.AND.M.NE.2) GO TO 11
0050        NEWWTH = WIDE-MAXO(WBOX(M),WBOX(1))
0051        SDIFF = WBOX(M)-WBOX(1)
```

```
0052      11 NEWLEN = NEWLEN+LBOX(M)
0053         IF(IABS(WBOX(M)-WBOX(1)).GT.DIFF) DIFF=IABS(WBOX(M)-WBOX(1))
0055       3 CONTINUE
0056         WIDE = NEWWTH
0057         GO TO 30
0058      25 NEWLEN = LENGTH
0059         NEWWTH = NEWWTH-WBOX(1)
0060         WIDE   = NEWWTH
0061      30 DO 4 P=1,N
0062         BOXLEN(P) = LLBOX(P)
0063       4 BOXWTH(P) = WWBOX(P)
0064         TPT     = POINT
         C*************************************************************
         C* ASSIGN THE UPPER LEFT CORNER OF ALL LOADED BOXES A       *
         C* COORDINATE (X,Y).                                        *
         C*************************************************************
0065         DO 5 Q=1,TRACK
0066         DO 6 B=1,A
0067         IF(MBL(B).LT.0) GO TO 6
0069         IF(MBL(B).NE.LBOX(Q).OR.MBW(B).NE.WBOX(Q)) GO TO 6
0071         MBL(B) = -MBL(B)
0072         COOR(B) = POINT
0073         GO TO 35
0074       6 CONTINUE
0075      35 POINT = POINT+100*LBOX(Q)
0076       5 CONTINUE
0077         POINT = TPT+WBOX(1)
0078         IF(SDIFF.GE.0) POINT = POINT+IABS(SDIFF)
0080         STPT  = POINT-WBOX(1)+100*LBOX(1)
0081         IF(DIFF.GT.4) GO TO 45
0083         IF(LENGTH-NEWLEN.GT.SMALL.OR.NEWWTH.LT.SMALL) RETURN
0085         NEWLEN = LENGTH
0086         GO TO 10
0087      45 RETURN
0088         END
```

```
      C*************************************************************
      C*                                                           *
      C* THIS SUBROUTINE IS USED IN CONJUNCTION WITH MAST3A AND    *
      C* PILE4.  THIS SUBROUTINE WILL TAKE THOSE BOXES IDENTIFIED  *
      C* BY TLBOX AND TWBOX, WHICH HAVE ALREADY BEEN LOADED, AND   *
      C* DELETE THEM FROM THE ORIGINAL LIST OF 'N' BOXES.  IT WILL *
      C* RETURN THE NEW LIST OF BOXES AS BOXLEN AN BOXWTH, WHICH   *
      C* CAN THEN BE USED TO CONTINUE LOADING THE PALLET.          *
      C*                                                           *
      C*************************************************************
0001        SUBROUTINE REGRP(BOXLEN,BOXWTH,MBOXL,MBOXW,TLBOX,TWBOX,
     *                       N,N1,N2)
0002        IMPLICIT INTEGER*2(A-Z)
0003        DIMENSION BOXLEN(30),BOXWTH(30),MBOXL(30),MBOXW(30),
     *            TLBOX(30),TWBOX(30),TL(30),TW(30),TTL(30),TTW(30)
0004        NN = 0
0005        MM = 0
0006        DO 1 P=1,N1
0007        DO 2 Q=1,N
0008        IF(MBOXL(Q).NE.TLBOX(P).AND.MBOXW(Q).NE.TWBOX(P)) GO TO 2
0010        MBOXL(Q) = -MBOXL(Q)
0011        GO TO 1
0012   2    CONTINUE
0013   1    CONTINUE
0014        DO 3 R=1,N
0015        IF(MBOXL(R).LT.0) GO TO 3
0017        NN = NN+1
0018        TL(NN) = MBOXL(R)
0019        TW(NN) = MBOXW(R)
0020   3    CONTINUE
0021        IF(N2.EQ.0) GO TO 10
0023        DO 4 S=1,N2
0024        DO 5 T=1,NN
0025        IF(TL(T).NE.-BOXLEN(S)) GO TO 5
0027        TL(T) = -TL(T)
0028        GO TO 4
0029   5    CONTINUE
0030   4    CONTINUE
0031        DO 6 U=1,NN
0032        IF(TL(U).LT.0) GO TO 6
0034        MM = MM+1
0035        TTL(MM) = TL(U)
0036        TTW(MM) = TW(U)
0037   6    CONTINUE
0038        DO 7 V=1,MM
0039        BOXLEN(V) = TTL(V)
0040        BOXWTH(V) = TTW(V)
0041        MBOXL(V)  = TTL(V)
0042   7    MBOXW(V)  = TTW(V)
0043        N=MM
0044        N2 = MM
0045        GO TO 15
0046   10   DO 8 W=1,NN
0047        BOXLEN(W) = TTL(W)
```